PERFORMANCE EVALUATION OF REAL-TIME DECISION MAKING ARCHITECTURES FOR COMPUTER INTEGRATED MANUFACTURING SYSTEMS

NAGW-1333

By:

Robert Y. Al-Jaar and Alan A. Desrochers

Department of Electrical, Computer, and Systems Engineering
Department of Mechanical Engineering, Aeronautical
Engineering & Mechanics
Rensselaer Polytechnic Institute
Troy, New York 12180-3590

CIRSSE Document #21
March 1989

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

1. INTRODUCTION

Computer Integrated Manufacturing (CIM) systems are interdisciplinary in nature and are heavily dependent on the acquisition, transmission and processing of data, information and knowledge. The efficient implementation of CIM requires the development of an integrated model of the overall system architecture and its control, communication, and database functions.

In its "Research Agenda for CIM: Information Technology" [1], the Panel on Technical Barriers to Computer Integrated Manufacturing states that the "Overall architecture for CIM deals with the basic question of how a large CIM system can be designed, developed, and modified in an orderly fashion. ... Common practice at the earliest stages of designing CIM systems is to make almost arbitrary decisions regarding hierarchical levels, the amount of central processing unit power and storage at each node, interconnection topology...". Designers of these architectures follow ad-hoc procedures requiring considerable amounts of experience, ingenuity and insight. Therefore, there is a great need to develop a modeling methodology to aid in the design and performance evaluation of such architectures for CIM systems. In this work, real-time Decision Making Architectures for CIM systems are modeled and evaluated. Although the accuracy of the decisions is very important, only the timeliness of the decisions is examined here.

1.1 What is a Decision Making Architecture for CIM Systems?

Generally, one thinks of an architecture as the configuration and interconnection of the "black boxes" that constitute the system. Johnson [2] defines architecture in the context of distributed control for CIM systems as "the choice and configuration of hardware and software modules which constitute the controller design". These control architectures operate in two distinct modes: <u>real-time</u> and <u>non real-time</u>. In the real-time mode, the control architecture responds to events that require immediate attention such as exception handling, error recovery, and other unscheduled events where the time scale is usually on the order of milliseconds to seconds. In the non real-time mode, the control architecture operates on a larger time scale ranging from minutes to hours or possibly longer. For example, the control architecture could be responsible for generating the production schedules for the next shift, issuing weekly preventive maintenance requests, and keeping updated records on statistical quality control.

In spite of their great importance, research on the topological aspects and temporal behavior of control architectures for manufacturing systems has attracted little attention. Related to such issues is the concept of a decision maker which evolved from research on humans as decision makers [3]. A mathematical model based on Petri Nets was used to analyze human decision making organizations for command and control applications [4-7]. Algorithms for the efficient generation and performance evaluation of admissible organizational architectures were developed, decision accuracy and organizational time delays were computed. In the context of CIM, Johnson discussed the requirements and defined the software modules of a typical node in a distributed hierarchical control architecture for automation [2]. However, no qualitative or quantitative analysis was done to evaluate the performance of alternative architectures. Scott et al. proposed a mathematical model to determine three-level hierarchical computer control requirements for a manufacturing system [8]. Costs were associated with the speed and memory of the computers. The configuration with the minimum cost was determined by solving a resource allocation problem using dynamic programming. No performance evaluation was done.

Therefore, to accurately model CIM architectures, the concept of a control architecture is broadened to include control, communication and database functions, resulting in an integrated real-time Decision Making Architecture (DMA). Control functions determine the logical flow of the information through the various processing stages of the DMA. Communication functions represent the physical flow of the information, while database functions represent the storage, retrieval, and updating of this information. This work does not address the actual algorithms that result in a decision, but rather evaluates the effect of a DMA's topology on its performance in a manufacturing environment.

Several modeling techniques such as Queueing Networks, Markov Chains, simulation, Perturbation Analysis and Formal Languages have been used to model manufacturing systems [9-14]. Modeling automated manufacturing systems using Petri Nets (PNs), Stochastic Petri Nets (SPNs), and Generalized Stochastic Petri Nets (GSPNs) results in integrated and unified models that are essential for the successful implementation of CIM systems [15-21]. PNs capture the basic <u>concurrent</u> and <u>asynchronous</u> behavior of manufacturing systems, and have several advantages over other methods, namely:

1. PNs can handle concurrency, capture asynchronous events, model logical precedence relations and represent structural interactions in a natural and simple way. Deadlocks, conflicts, and finite buffer sizes in a manufacturing system can be easily modeled and efficiently analyzed.

2. PN models represent a hierarchical modeling tool with a well-developed mathematical and practical foundation. Structural (deadlocks, mutual exclusion, liveness and boundedness), and temporal analyses (performance measures such as throughput rate, resource utilizations and in-process inventory) can be carried out using GSPNs.
3. Since SPNs are isomorphic to Markov Chains [22-23], and GSPNs to Embedded Markov Chains [24], using them does not require a deep knowledge of stochastic systems. PNs address the complexity issue associated with the modeling of manufacturing systems. Simple PN models can represent very complex Markov Chains.
4. Their graphical nature helps to visualize such complex systems. Software packages have been developed which automatically generate and solve the Markov Chain from the GSPN. This frees the modeler from having to painstakingly account for all possible states of the system.
5. Incremental changes to a PN model are done easily, while minor changes in a Queueing Network or Markov Chain model require, in most cases, altering the whole model.
6. Finally, PN models can also be used to implement real-time control systems [25].

2. OBJECTIVES AND ORGANIZATION

The objectives of this research are:

1. To develop a generic modeling methodology with a flexible and modular framework to aid in the design and performance evaluation of automated manufacturing systems using a unified model.
2. To identify and quantify some of the information requirements for the design of architectures for CIM systems by modeling the logical and physical flow of the information through the various processing and storage stages.
3. To demonstrate the applicability of this methodology by modeling and evaluating several hierarchical real-time decision making architectures.

In accordance with these objectives, such a modeling methodology is proposed here, whose main features are:

1. **Answer "what-if" Questions.** The proposed methodology provides answers to many "what-if" questions? e.g. What-if the database access time is increased? What-if communication delays are decreased?
2. **Modularity and flexibility**. In spite of apparent contradiction, the proposed methodology ensures modularity and flexibility. Basic building blocks are defined and used to model both hardware and software components, and can be customized to facilitate the modeling and evaluation process.
3. **Applicability to Discrete Event Dynamic System modeling.** CIM systems are event-driven, and the proposed methodology captures their concurrent and asynchronous behavior.

4. **Availability of analysis methods.** The same model supports structural and temporal analyses.
5. **Graph-based.** To manage the size and complexity of these systems, the proposed methodology is graph-based with a potential to support object-oriented programming techniques, and iconic libraries that hide the details of the modeling methodology.

To choose the computer hardware and software needed to control the factory, the system designer can use this proposed methodology to answer "what-if" questions that affect many design choices such as:

* How many computers are needed and how should they be interconnected?
* What is the required processing power for each computer and is there a need for multiprocessing capability?
* How much local memory and storage capacity should be allocated to each computer?
* What are the needed software modules, and how should they be allocated among the computers?
* Is there a need for a dedicated communication line or will a Local Area Network be required? What Local Area Network topology and protocol should be used?
* What is the most appropriate database model and what is the best storage, retrieval and update policy? Is there a need for a local database, centralized, or distributed database?

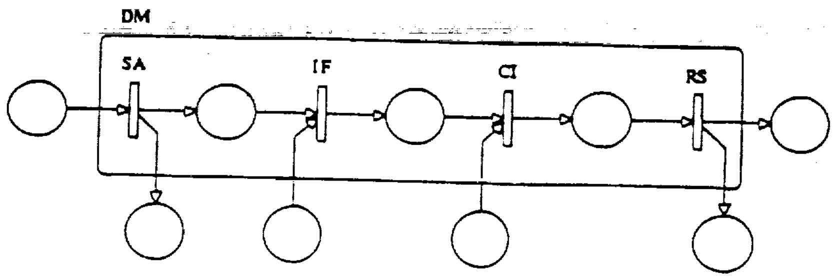
Section 3 presents the GSPN model of a Decision Making Unit (DMU). Section 4 explains the modeling assumptions. The performance evaluation of an isolated DMU is done in section 5. Section 6 explains how to compute the response time. Sections 7-9 evaluate the performance of three real-time decision making architectures: two DMUs arranged in two levels, four DMUs arranged in two levels, and six DMUs arranged in three levels, respectively. Section 10 demonstrates the use of the proposed methodology as a design tool. Model reduction and approximation is briefly explored in section 11. Section 12 summarizes the main results of this work. This work concludes with a discussion of future research.

3. GSPN MODEL OF A DECISION MAKING UNIT

The main purpose of a real-time DMA in a CIM system is to ensure the accurate execution of a given production plan, and detecting errors and correcting for any deviations from that plan [2]. The architecture should respond in a timely and accurate manner to any event that requires its decision making capabilities.

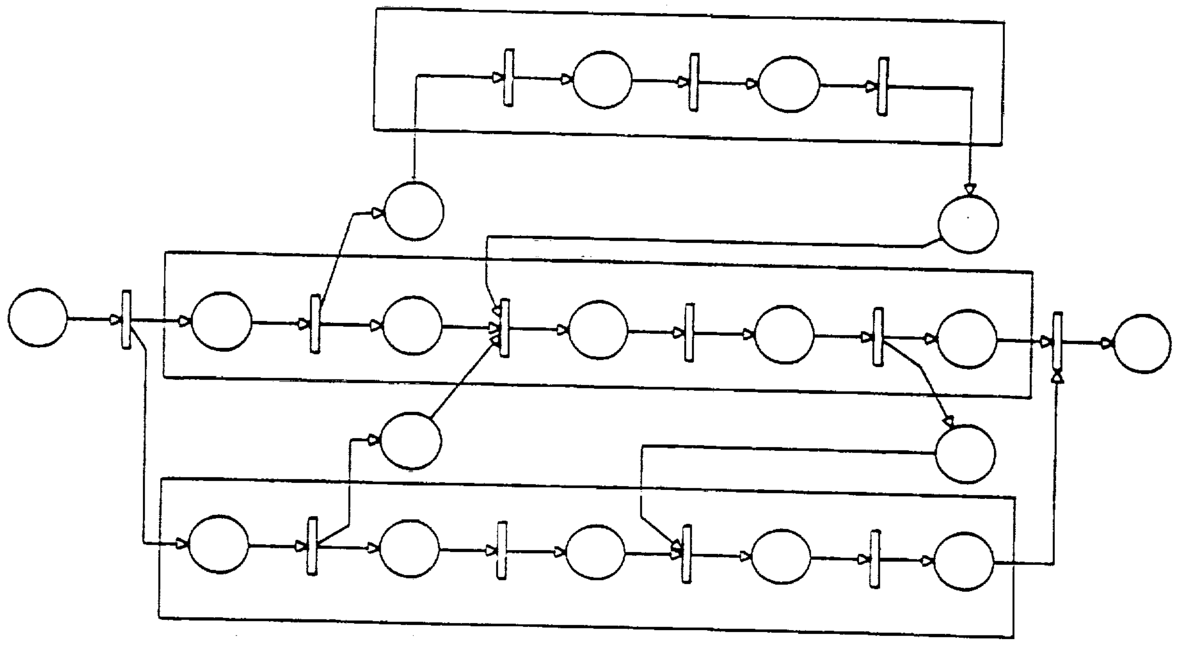
The decision maker suggested in [4-6] consisted of four processing stages connected in series as shown in Figure 1a. The algorithms in the Situation Assessment stage process the input signals. Information from other decision makers is merged using the algorithms in the Information Fusion stage. The Command Interpretation stage interprets external commands by combining them with internal information. Finally, a response is selected by the algorithms in the Response Selection stage. Figure 1b shows a simple model of a human decision making organization.

This model is not appropriate in a manufacturing context for the following reasons:



SA: Situation Assessment
IF : Information Fusion
CI: Command Interpretation
RS: Response Selection

a) Petri Net model of a four-stage decision maker.



b) Decision making organization.

Figure 1: Petri Net model of a four-stage decision maker and organization [6].

1. In a real-time DMA for CIM systems, one is interested in determining the processing power of each computer, the allocation of functions and databases to each computer, and the choice of hierarchical levels. The decision maker does not provide the needed mechanism to model and evaluate these choices.
2. The decision maker organization contains no information loops. Controlling manufacturing operations makes extensive use of feedback. Modeling communication and database functions cannot be done with acyclical models.
3. The decision maker does not model the functions of the computers encountered in a real time integrated manufacturing environment. Explicit models of the control, communication, and database functions should be incorporated.
4. Basic characteristics of manufacturing systems such as component failures, conflicts, resource sharing, and schedule priorities cannot be modeled using the decision maker since it does not support places with multiple arcs.
5. The decision maker organization has only one input and one output. The decision makers are tightly coupled where <u>all</u> of them respond to the <u>same</u> stimulus. In a manufacturing system, each computer controls a segment of the factory and interacts with the process and/or other computers independently of others.

A typical node in a real-time DMA for manufacturing systems consists basically of a computer, software, memory or database, and communication interfaces, and/or sensors and actuators. To achieve the desired performance, each node must accomplish several goals as outlined in Figure 2 [2]. The control computer sends the processed sensory information to other computers. It also interprets the commands received from higher level nodes, and sequences them for use by other (lower) level nodes. As needed, the node could access a database that contains an up to-date model of a segment of the factory floor.

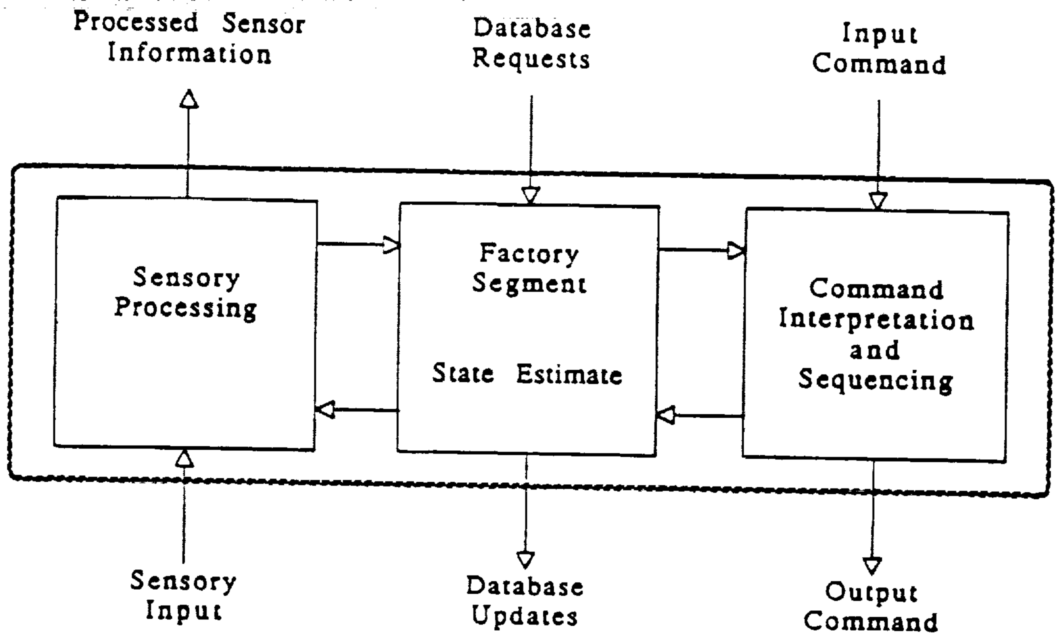


Figure 2: Functions of a typical node in a manufacturing control system [2].

Based on the requirements of a real-time DMA and the functional description of a typical node in a CIM system, a GSPN model of a typical Decision Making Unit

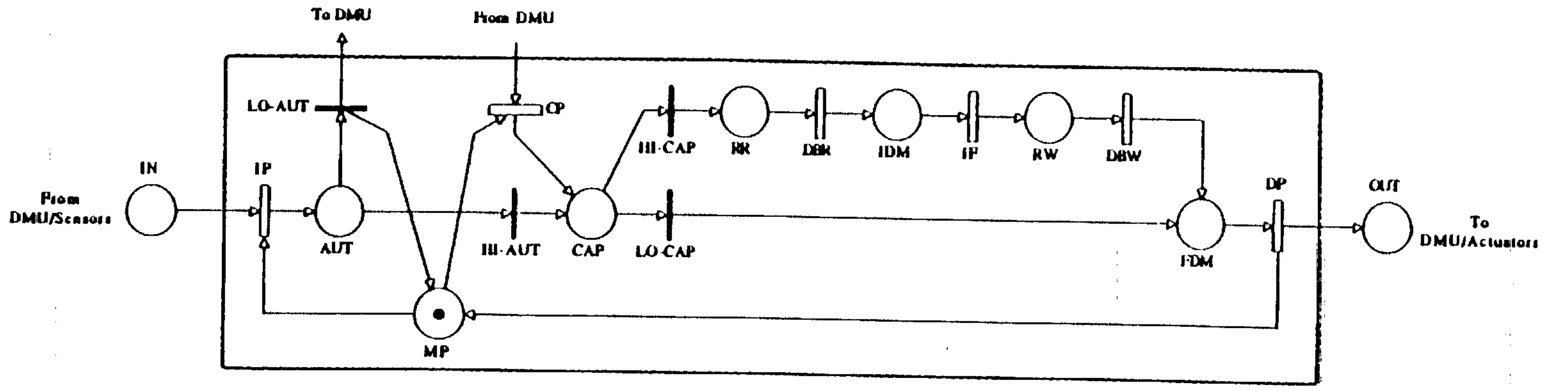
(DMU) is proposed in Figure 3. Although it has been motivated by the decision maker of Figure 1, the DMU is designed to model situations that are unique to a CIM environment. This basic DMU module will serve as a building block for evaluating several real-time DMAs. The decision maker in Figure 1 consisted only of timed transitions, while the DMU in Figure 3 has timed transitions drawn as white boxes, and immediate ones drawn as black bars.

3.1 Response Time as a Measure of Performance

Time imposes a major constraint at all levels of manufacturing. The timely acquisition, storage, processing, retrieval, and transmission of data, information, and knowledge in a CIM system is critical to its proper operation. Therefore, the main measure of performance used in this work is the response time (RT) of the DMA.

A decision constitutes a response to a request. The response time *is the elapsed time between the instant a request for intervention is sent by (part of) the manufacturing process to the DMA and a response from the DMA is received.* A DMA with a "large" RT might be incapable of promptly correcting for detected errors. Depending on the situation at hand and the interpretations of the model, RT could measure the time a workstation spends waiting to receive further instructions when one of its machines fails or its buffer is full or the time it takes for a machine to receive an NC program, or the time the manufacturing process must wait until a new production schedule is obtained.

Performance measures such as the average number of queued requests or responses, and the utilization or workload of a DMU can also be used. All of these



CP: Command Processing
IP: Input Processing
DBR: Data Base Read
DBW: Data Base Write
IF: Information Fusion
DP: Decision Processing

Figure 3: GSPN model of a typical Decision Making Unit.

measures are related to each other. For example, an increase in the average number of queued requests results in an increase in RT, so will an increase in the workload. These measures of performance quantify the temporal behavior of a DMA. Other non-temporal measures are of equal importance. For example, the accuracy of the decisions, and the consistency and correctness of the data need to be quantified and evaluated. However, as stated earlier, this is not the focus of the current work.

For a given DMA, the response time RT is evaluated as a function of several variables, namely, the Decision Making Rate, the Degree of Autonomy, and the Decision Making Capacity of a DMU, as well as the Degree of Cooperation among DMUs.

The Decision Making Rate is the number of decisions a DMU can make per time unit, and is mainly dependent on hardware. For example, it can be increased by adding a multiprocessing capability to the DMU.

The Degree of Autonomy represents the percentage of the requests that a DMU can respond to locally, without the need to ask another DMU for intervention. Autonomy is dependent on software, and can be increased by allocating more functions to the DMU using software capable of handling a wider variety of situations.

The Decision Making Capacity reflects the percentage of the requests that require database accesses and information fusion.

The Degree of Cooperation indicates the percentage of responses received from a parent DMU. This is a measure of peer cooperation since a request received by the parent DMU could originate from any of its children DMUs.

The system designer has usually more control over the Decision Making Rate and the Degree of Autonomy, than over the Decision Making Capacity and the Degree of Cooperation which are determined by the actual manufacturing process. The following example illustrates how a given scenario can be quantified using the above variables. Given a manufacturing process that is controlled by a two-level hierarchy of computers where four identical workstations are connected to one cell. From previous experience it is known that responding to the occurrence of 40 out of 50 potential events requires a large amount of data and information fusion algorithms. This system can be modeled as a two-level DMA with five DMUs. From the knowledge of possible events, the Decision Making Capacity is determined to be 80%. Identical workstations imply that the cell controller could assign a task to any one of them with equal probability. Therefore, the Degree of Cooperation is 25%. The designer can now evaluate the performance of the DMA as the Decision Making Rate, and the Degree of Autonomy varies. Assume that analysis results showed that the "best" performance is obtained with a 60% Degree of Autonomy and a Decision Making Rate of 5 decisions per time unit for the workstations. The designer concludes that the workstations must have the necessary software to enable them to locally respond on average to 30 out of the 50 events. Also, if collected data indicated that the Decision Making Rate of each workstation was less than 5, then new improved hardware might be needed to achieve the desired performance. This is a simple example since it does not consider possible delays due to distributed database management or communication network protocols. However, it helps to explain the underlying ideas that are needed to understand the models and results of this work.

3.2 Model Interpretation

The DMU in Figure 3 is composed of 1) random switches modeled as immediate transitions representing probabilistic choices among several alternatives, and 2) several processing stages modeled as exponentially timed transitions representing the time it takes to execute the algorithms associated with them. The marking of the places indicate the state of the DMU. Table 1 gives the interpretation associated with each of these transitions and places.

Transitions

IP	Input Processing stage (timed).
CP	Command Processing stage (timed).
HI-AUT	High Autonomy (immediate).
LO-AUT	Low Autonomy (immediate).
HI-CAP	High decision making Capacity (immediate).
LO-CAP	Low decision making Capacity (immediate).
DBR	Data Base Read stage (timed).
IF	Information Fusion stage (timed).
DBW	Data Base Write stage (timed).
DP	Decision Processing stage (timed).

Places

IN	Input is available from other DMUs or physical sensors.
CMD	Command or response from other DMUs.
AUT	Choice of the DMU's degree of autonomy.
CAP	Choice of the DMU's decision making capacity.
RR	Request for a database read operation.
IDM	Intermediate decision making.
RW	Request for a database write operation.
FDM	Final decision making.
OUT	Response to other DMUs or physical actuators.
MP	Multiprocessing capability of a DMU.

Table 1: Interpretation of transitions and places of a DMU.

The HI-AUT and LO-AUT immediate transitions model the Degree of

Autonomy of a DMU using a random switch with prob{HI-AUT} = hi-aut and prob{LO-AUT} = lo-aut. A higher value for hi-aut implies more decisions are made locally. Note that hi-aut + lo-aut = 1.

The <u>HI-CAP</u> and <u>LO-CAP</u> immediate transitions model the Decision Making Capacity of a DMU using a random switch with prob{HI-CAP} = hi-cap and prob{LO-CAP} = lo-cap. A smaller value for hi-cap implies that the DMU is capable of making more simple than complex decisions. Note that hi-cap + lo-cap = 1.

The <u>IP</u> stage processes the information received from other DMUs or from a physical sensor such as a bar code reader, requiring minimal processing, or a camera, requiring substantial processing. In the latter case, the IP stage could model the time it takes the camera preprocessor to send the information to the DMU.

The algorithms in the <u>CP</u> stage receive high level commands from higher level DMUs in a hierarchical architecture, interpret, and sequence them into the appropriate set of lower level primitive commands. For example, a cell node issues a command to the AGV workstation node to "Move AGV #12 to Workstation #5".

The <u>DBR</u> and <u>DBW</u> stages model the time it takes to read and update the databases. These databases contain a model of the factory, or a segment of it, that allow the DMU to carry out its tasks. For example, in order to determine if a resource shared with another cell is available, the DMU queries the database about the status of that resource. The cell node might also need to update the database model so that the correct information be available to other nodes. The databases could be local, shared, distributed, etc... as defined and modeled by the designer. In this work, mainly dedicated databases are considered and the existence of some mechanism that maintains the consistency of the databases is assumed.

The IF stage fuses the information received from other DMUs, sensors, and databases. For example, the algorithms in the IF stage could be responsible for processing the camera frames of a robot in motion along with information from the data base to plan a collision free path of the robot.

The DP stage models the need for specialized final processing before sending the output to other DMUs or the physical actuators. For example, there could be a need to process the commands further since a particular machine might have a non-standard hardware interface.

All the places represent the status of the DMU and are self-explanatory with the exception of MP which models the multiprocessing capability of a DMU. Since new computer technology allows for multiprocessing, the place MP is added for flexibility. In most of this work, however, it is assumed that each DMU can process only one request at a time, which is modeled by initially having only one token in MP. Eliminating MP will, in effect, give the DMU unlimited multiprocessing capability.

It is important to note here that the above model and interpretations are designed to provide for a high degree of modularity and flexibility. In fact, the designer might wish to replace some of the timed transitions by immediate ones if they are seen as less important or having negligible impact on the system's performance. Also, the designer might elect to eliminate some stages altogether. Figure 4 demonstrates this flexibility by showing several possible variations on the GSPN model of a DMU. The

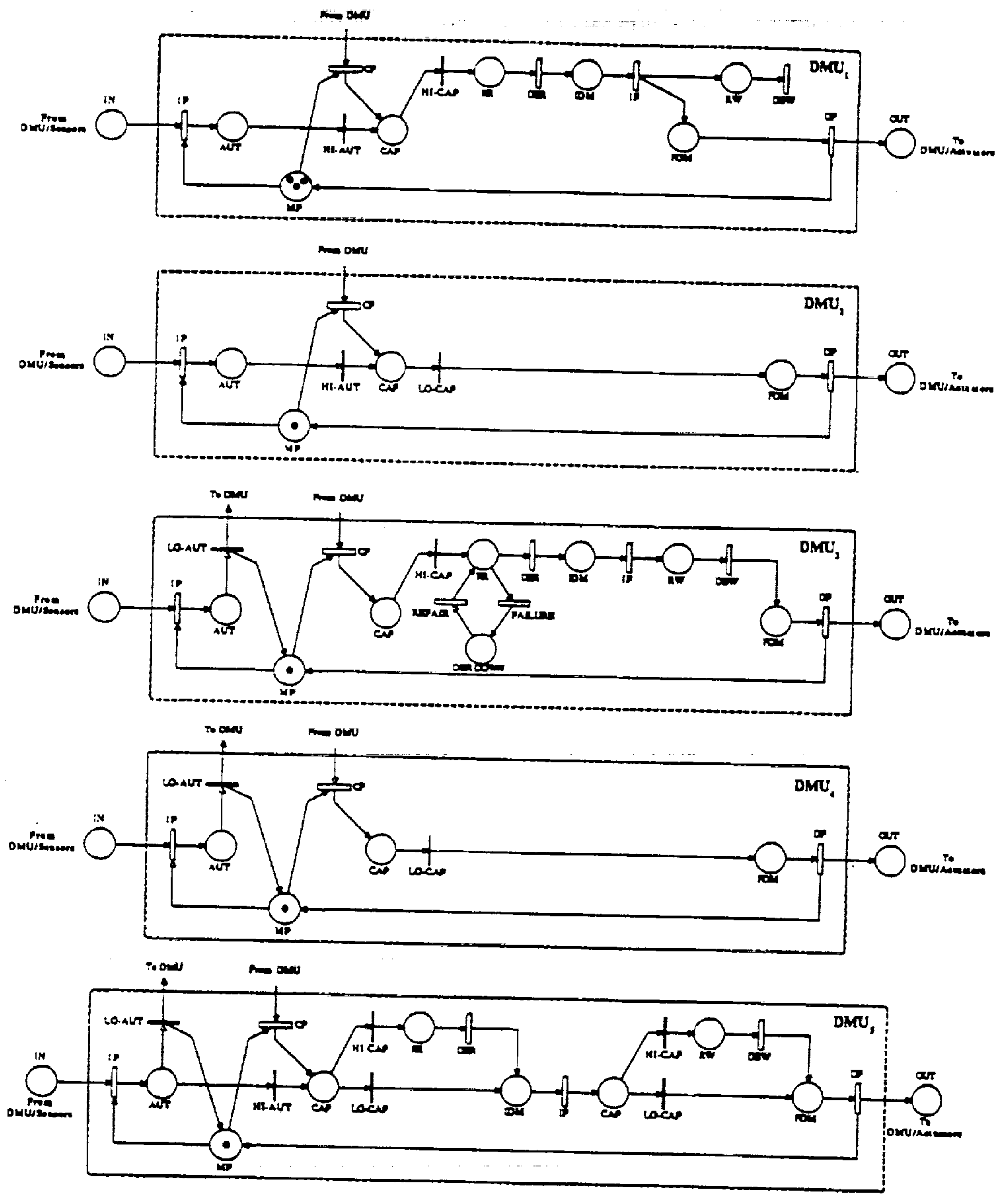


Figure 4: Variations on the GSPN model of a DMU.

differences between these customized DMUs and the one in Figure 3 are:

DMU_1	Following the IF stage, the DMU proceeds with its decision making process while the appropriate database updates take place concurrently. It can also process three requests simultaneously.
DMU_2	Completely autonomous without the need for database accesses and information fusion.
DMU_3	Although no decision is made locally, the interpretation and sequencing of all the higher-level commands require database accesses and information fusion. Also, the model takes into account the failure and repair of the DBR operation.
DMU_4	No decision is made locally, and no database accesses and information fusion capabilities are needed.
DMU_5	The strict ordering of DBR followed by DBW is relaxed. A DBR is not necessarily followed by a DBW, and a DBW is not necessarily preceded by a DBR.

Table 2 explains all the acronyms used in this work.

AGV	Automatic Guided Vehicle.
CIM	Computer Integrated Manufacturing.
DMA	Decision Making Architecture.
DMU	Decision Making Unit.
FIFO	First-In First-Out.
GSPN	Generalized Stochastic Petri Net.
PN	Petri Net.
PRO	The physical manufacturing PROcess.
RT	Response Time.
SPN	Stochastic Petri Net.

Table 2: List of acronyms.

3.3 DMU Classification

O'Grady [26] qualitatively classified the cells in an automated manufacturing system into four types based on their decision making ability and local memory access. Within the framework of the proposed DMU model, these can be quantified through the use of random switches representing the Degree of Autonomy, and the Decision Making Capacity of the DMU. For the Degree of Autonomy, **High AUT** is modeled using hi-aut > 0.5 and **Low AUT** using lo-aut > 0.5. Similarly for the Decision Making Capacity CAP. The four types are:

1. **High AUT/High CAP.** This represents a DMU that makes most of its decisions locally and requires heavy access to databases and information fusion algorithms. For example, a cell computer controlling two co-operating robots must have a high degree of autonomy and decision making capacity due to the complex task of motion planning and controlling the robots. Also, error recovery and exception handling are important to ensure a safe operation. DMU_1 of Figure 4 is an extreme case of this class.
2. **High AUT/Low CAP.** This models a DMU capable of making most of its decisions locally without requiring information fusion or frequent database accesses. For example, the microprocessor controlling a bar code reader, a laser ranger and a camera could have the capability to locally resolve most of the discrepancies and conflicting sensory information without the need for frequent database accesses or information fusion. DMU_2 of Figure 4 is an extreme case of this

class.

3. **Low AUT/High CAP.** This represents a DMU capable of making only a few local with considerable need of databases and information fusion. For example, a cell computer that controls several AGVs could have the ability to plan the needed shortest collision free paths and allocate the shared tracks accordingly. However, it must receive commands about the destination of each AGV from a shop controller that plans the allocation of AGVs for the entire shop floor. DMU_3 in Figure 4 is an extreme case of this class.
4. **Low AUT/Low CAP.** This models a DMU capable of locally making only a few decisions, requiring minimal amounts of information fusion and infrequent database accesses. For example, a terminal used as an operator console does not have a need for a high degree of autonomy or complex decision making ability. DMU_4 in Figure 4 is an extreme case of this class.

3.4 Correct Use of Random Switches

The use of random switches increases the modeling power and adds to the flexibility of the proposed methodology. Their proper use is essential to obtaining correct models with meaningful performance measures. The probabilities associated with the immediate transitions of a random switch affect the state transition rates of the Embedded Markov Chain, and hence, the steady-state probabilities. It is possible that more than one set of transitions, belonging to more than one random switch, be enabled in a given marking. This may result in the generation of an Embedded Markov Chain that does not correctly model the desired behavior of the system. Figure 5 shows a portion of some GSPN model containing two random switches. With $m(RS_1) = 1$ the probability distribution of its transitions is (0.2, 0.3, 0.5). With $m(RS_2) = 1$ the probability distribution of its transitions is (0.4, 0.6). With $m(RS_1) = m(RS_2) = 1$ the probability distribution of the five transitions is (0.1, 0.15, 0.25, 0.2, 0.3). It would seem that to ensure the correct use of immediate transitions one must first generate the Reachability Graph and then assign the appropriate probabilities. This is impractical. Techniques and guidelines on the correct use of random switches were suggested in [27-28]. They enable the modeler to correctly define the discrete probability distributions at the net level without the need to generate the Reachability Graph. Accordingly, the DMU module was designed to guarantee that, in a DMA model, the following properties hold:

1. Each of the Extended Conflict Sets is a free-choice set.
2. All Extended Conflict Sets are mutually exclusive.

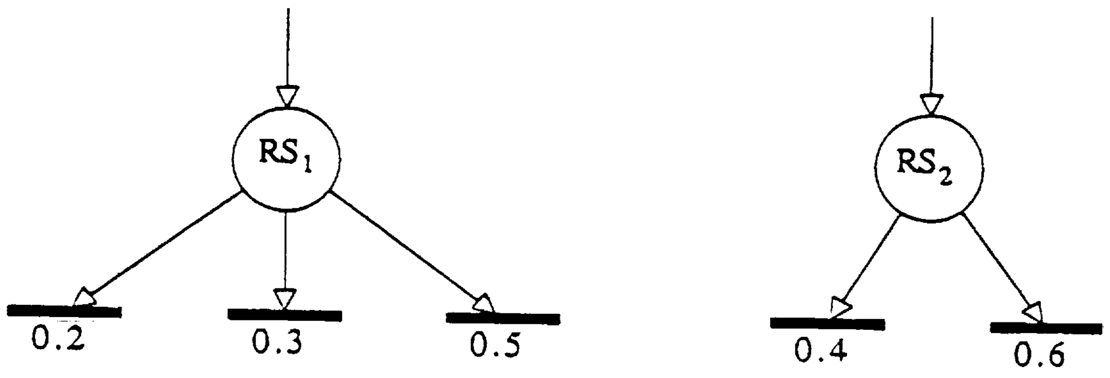


Figure 5: Selection of discrete probability distribution for random switches.

An Extended Conflict Set can be thought of as a set of possibly conflicting (immediate) transitions. Property 1 implies that all the probabilities of the random switches can be defined locally at the net level. Property 2 implies that there need be only one priority level assigned to all the immediate transitions which should be higher than that for the timed ones. If the modeler wishes to distinguish between the transitions of various Extended Conflict Sets different priorities may still be used.

4. MODELING ASSUMPTIONS AND EXPERIMENTAL SETUP

Modeling large complex systems such as real-time DMAs for manufacturing systems is a difficult task that requires the modeler to impose certain assumptions in order to obtain tractable models. The modeling assumptions in this work are:

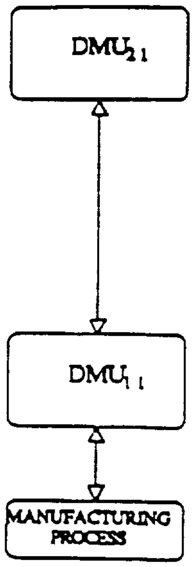
1. The times associated with the transitions are exponentially distributed random variables. Then, for a live, bounded, and reversible GSPN, the isomorphic Embedded Markov Chain can be solved for the steady-state probabilities.
2. Each process can issue one and only one request for intervention at a time and must wait for the response to a previous request before initiating another one.
3. Each part of the manufacturing process connected to the lowest level DMUs is modeled as an immediate transition. This will be justified in the next section.
4. The rates of the timed transitions in a given DMU are arbitrarily selected to be equal.

5. The structure of all the DMUs and their interconnections are chosen to be identical. In concurrence with hierarchical control practices, no direct interactions among peer DMUs are allowed.
6. Each DMU has a dedicated database. A mechanism for updating <u>all</u> the databases of the DMUs is assumed to exist, but is not explicitly modeled in this work.
7. Communication networks are not explicitly modeled. It is assumed that whenever a physical link is needed, it will be made available. Integrating communication network models can be easily done by breaking off the appropriate arcs among the DMUs and inserting the desired PN model.

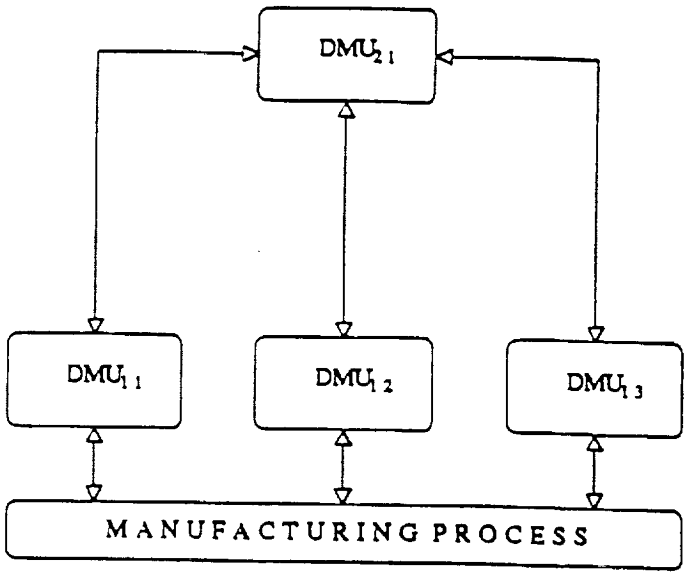
Note that the above assumptions can be relaxed as desired, including the exponential distribution associated with the firing rates.

Several experiments will be carried out to evaluate and compare the performance of alternative hierarchical architectures. STAR-1 consists of two DMUs arranged in two levels, STAR-3 consists of four DMUs arranged in two levels, and HIER-3 consists of six DMUs arranged in three levels. Figure 6 shows a block diagram of these configurations. The detailed GSPN model of two interacting DMUs is shown in Figure 7. Future DMA models will be based on these interactions, which result in live, bounded, and reversible GSPNs. Since these models are too large to clearly fit on a single page, future figures will be shown without labels on the places and transitions.

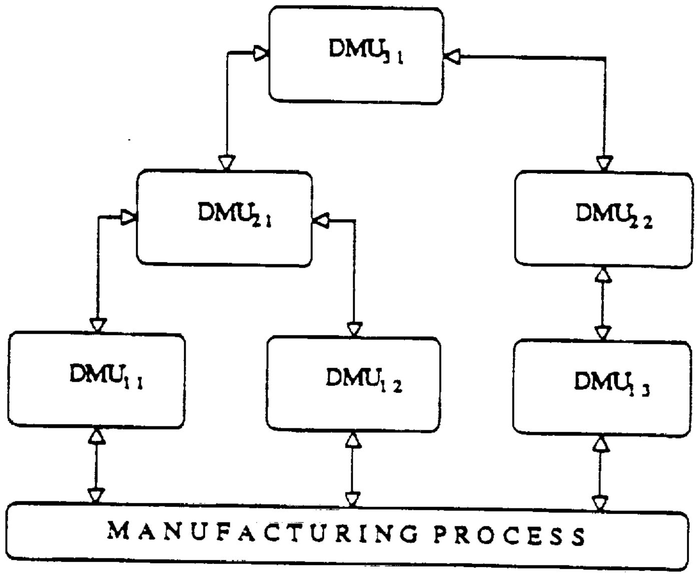
To carry out these experiments, the following numerical values were chosen:



STAR-1



STAR-3



HIER-3

Figure 6: Examples of hierarchical real-time Decision Making Architectures.

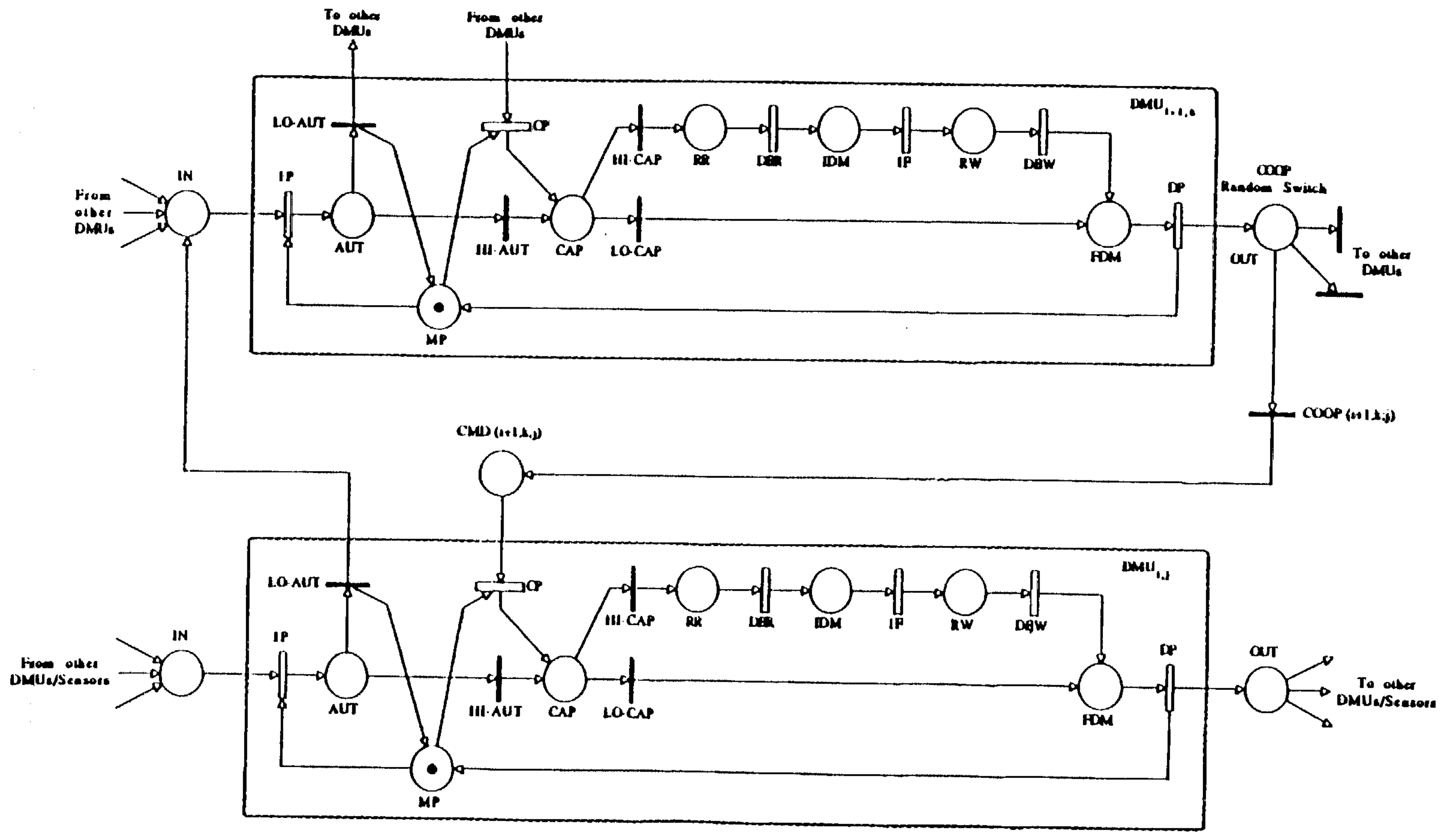


Figure 7: Interactions between two DMUs.

dmu-rate = 1 decision per time unit,
hi-aut = lo-aut = hi-cap = lo-cap = 0.5,
m(MP) = 1 token for all DMUs,
m(IN) = 1 token for the lowest level DMUs,
equal probability of cooperation among DMUs.

These values represent the standard values that will be used when not explicitly stated otherwise. When varying hi-cap, three values will be used, namely, (0.1, 0.5, 0.9).

Finally, it is important to make clear that this work is concerned with evaluating the real-time response of these architectures. It is common when controlling a manufacturing facility to take advantage of the different time scales at the various levels of the hierarchy. For example, in the NIST hierarchy [29], a shop floor computer may be in-charge of planning and scheduling part of the factory over an eight-hour shift. This is a non real-time operation. On the other hand, if the lowest level computer, say at the equipment level, encounters an event that it cannot handle, it usually requests the intervention of the next level computer, say the workstation controller. If the workstation controller cannot resolve the issue, it will request the intervention of the cell controller, and so on. Hence, in response to some event that occurs in the manufacturing process, the intervention of the shop floor computer might be needed. Although the normal operations at that level have a time scale on the order of an eight-hour shift, the shop controller must respond immediately to that event. The evaluation of this real-time performance is discussed here.

5. ANALYSIS OF AN ISOLATED DMU

In order to gain some insight into the performance of DMAs, an isolated DMU

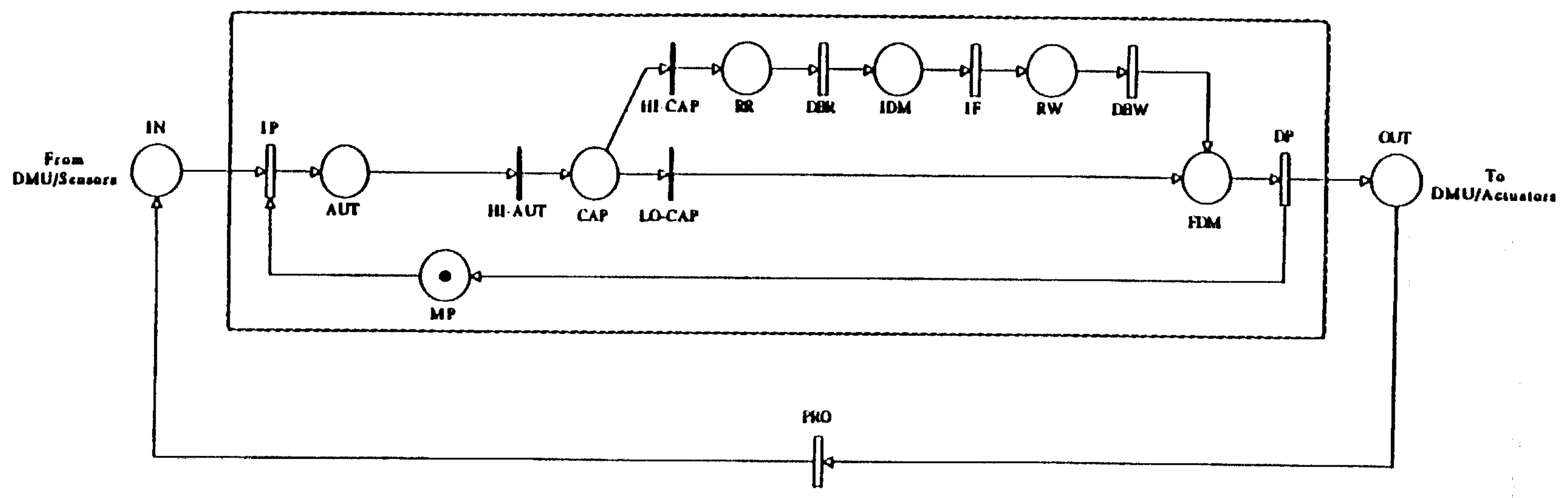


Figure 8: GSPN model of an isolated DMU.

is analyzed first. Figure 8 shows the DMU connected to a manufacturing process PRO. PRO could represent a single machine, such as a lathe, controlled by one computer. It could also represent a complex setup of several machines controlled by a centralized computer. In fact, the macro transition PRO could be a complex PN model such as a transfer line or a production network [18,20]. PRO requests the intervention of the DMA through the sensors hardware and receives a response from the DMA through the actuator's hardware. These requests may originate from the various components modeled in PRO such as machines, buffers, and AGVs.

5.1 Structural Analysis

The invariants give an insight into the structural properties of the DMA. There are two basic P-invariants and two basic T-invariants for this model:

P-invariants:

$$m(IN) + m(AUT) + m(CAP) + m(RR) + m(IDM) + m(RW) + m(FDM) + m(OUT) = 1,$$
$$m(AUT) + m(CAP) + m(RR) + m(IDM) + m(RW) + m(FDM) + m(MP) = 1.$$

T-invariants:

{IP, HI-AUT, HI-CAP, DBR, IF, DBW, DP, PRO},
{IP, HI-AUT, LO-CAP, DP, PRO}.

The P-invariants indicate that at any moment the DMA is engaged in at most one activity. This is expected since there is only one process and one DMU. The second P-invariant is a consequence of the MP place and eliminating it leaves only the first P-invariant.

The T-invariants show the decision paths needed to process a request. PRO receives a response through two different paths. The first is the HI-CAP path indicated by the first T-invariant. The second T-invariant reflects the LO-CAP path. RT will measure the total average times of these paths weighted by the hi-aut and hi-cap probabilities, reflecting the Degree of Autonomy, and the Decision Making Capacity, respectively.

5.2 Performance Evaluation

The rate of the timed transitions are all equal to dmu-rate, while PRO has a rate equal to pro-rate. It will be shown that the computation of RT is independent of the value of pro-rate. The DMU is completely autonomous with hi-aut = 1. RT is the time it takes a token to go from IN to OUT. Since the net is safe and the average steady state flow of tokens through IP, DP and PRO is equal (they belong to the same T-invariants), the response time is

$$\begin{aligned} RT &= 1/[\text{dmu-rate prob}\{\text{IP is enabled}\}] - 1/\text{pro-rate} \quad \text{time units,} \\ &= 1/[\text{dmu-rate prob}\{m(IN)\, m(MP) = 1\}] - 1/\text{pro-rate} \qquad (1). \end{aligned}$$

The log-log scale in Figure 9 shows that RT is inversely proportional to dmu-rate, or directly proportional to the decision time of DMU, and is also directly proportional to the Decision Making Capacity of the DMU as in Figure 10.

For this simple case, one may compute a closed-form expression for RT. Figure 11a shows the Reachability Graph for this GSPN model which has 6 tangible and 2 vanishing markings. Marsan [24,27] describes a method to obtain the steady-state probabilities of the markings by noting that the Reachability Graph is isomorphic to a Stochastic Point Process with a finite state-space, and that an

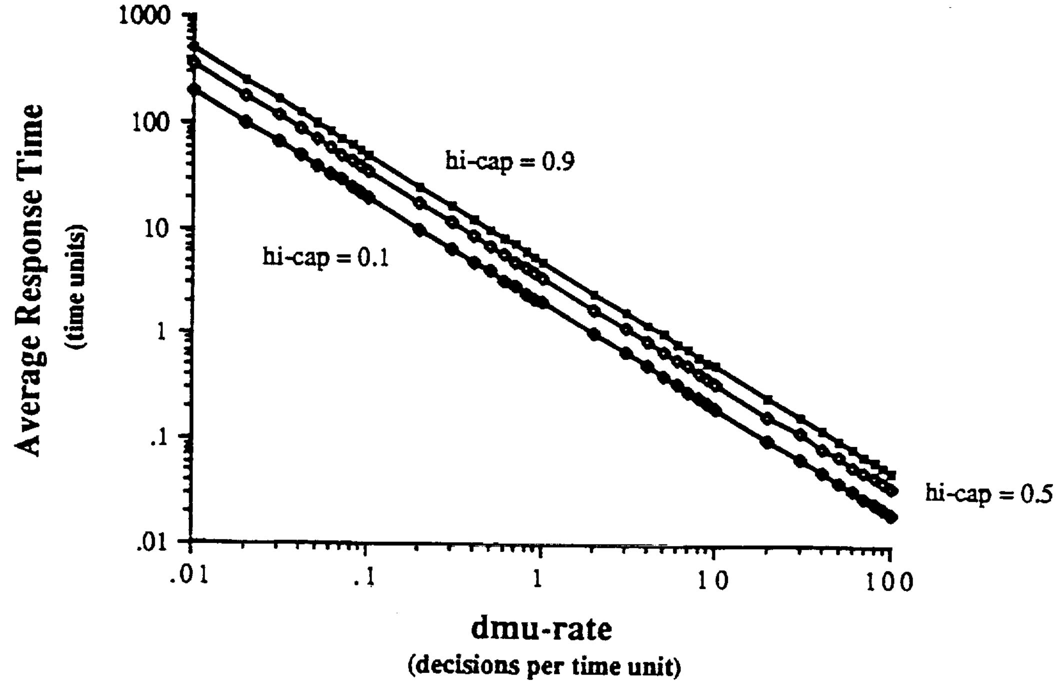


Figure 9: RT vs. dmu-rate of an isolated DMU.

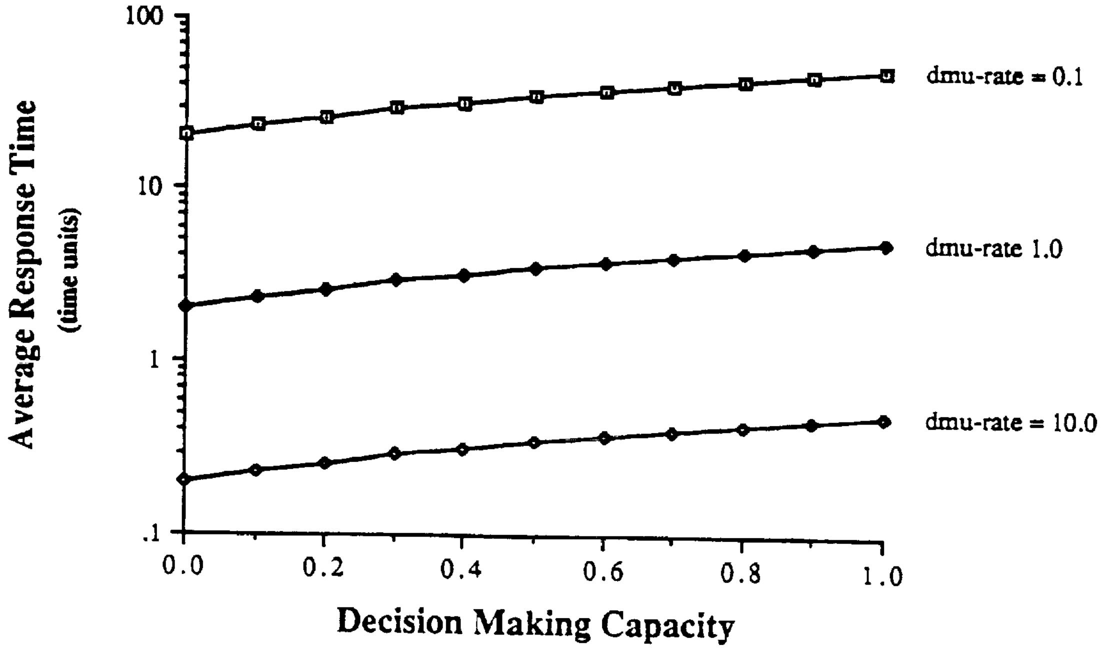


Figure 10: RT vs. hi-cap of an isolated DMU.

Embedded Markov Chain can be recognized within this process. The desired probabilities can then be computed from the solution of the Embedded Markov Chain. For this simple example, an equivalent Markov Chain can be written by inspection as in Figure 11b. Let π_i represent the steady state probability of being in state i. Then, the balance equations of the Markov Chain are:

$$\begin{aligned} \text{dmu-rate}\ \pi_0 &= \text{dmu-rate}\ \pi_4 = \text{pro-rate}\ \pi_5, \\ \text{dmu-rate}\ \pi_1 &= \text{dmu-rate}\ \pi_2 = \text{pro-rate}\ \pi_3, \\ \text{hi-cap dmu-rate}\ \pi_0 &= \text{dmu-rate}\ \pi_1, \end{aligned}$$

where the sum of all probabilities equals one.

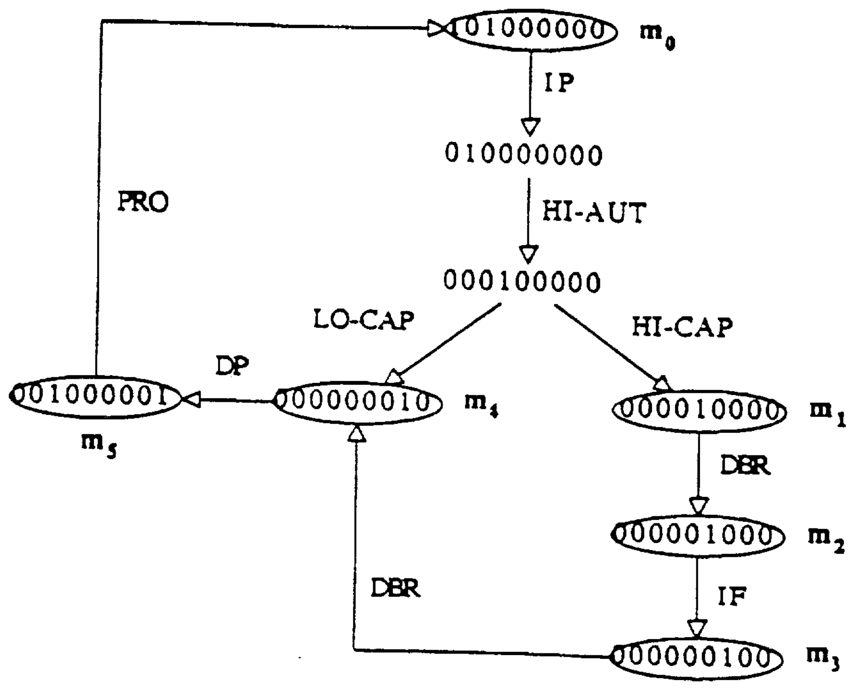
Solving for π_0

$$\begin{aligned} \pi_0 &= \text{prob}\{m(IN)\ m(MP) = 1\} \\ &= 1/(2 + 3\ \text{hi-cap} + \text{dmu-rate/pro-rate}) \qquad (2). \end{aligned}$$

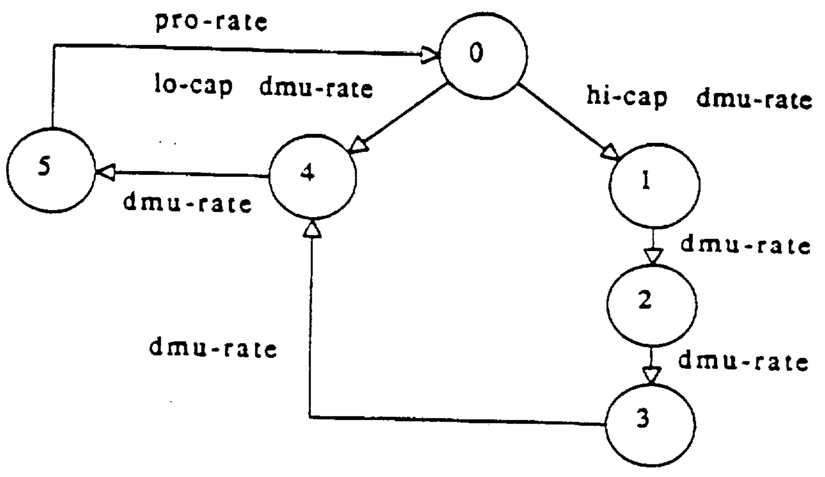
Substituting (2) into (1) the response time becomes

$$\begin{aligned} RT &= (2 + 3\ \text{hi-cap})/\text{dmu-rate} \quad \text{time units} \qquad (3), \\ &= (2 + 3\ \text{hi-cap})\ \text{dmu-time} \quad \text{time units} \qquad (4). \end{aligned}$$

When holding one of the variables in (3) constant, RT becomes a linear function of the time it takes a DMU to make a decision, and of its Decision Making Capacity, in agreement with Figures 9-10.



a) Reachability Graph.



b) Equivalent Markov Chain.

Figure 11: Reachability Graph and equivalent Markov Chain for the isolated DMU.

Finally, recall that RT is defined as the average time it takes for a token to go from place IN to place OUT by traveling over the two paths indicated by the T-invariants but without passing through PRO. Equation (3) shows that, in fact, RT is independent of PRO. This is desired since the time it takes for a DMA to respond to a request should not depend on how long the process takes to act on the response. The assumption made in section 4 to model PRO as an immediate transition is therefore justified.

6. COMPUTING THE RESPONSE TIME

If the net is not safe or the transition of interest is connected to unsafe places, the above method (which will be called the throughput method) for computing RT cannot be used since the tokens will interfere with each other. Little's Law should be used instead [30]. It states that the average delay of a customer in a "system" is equal to the average number of customers in that system divided by the effective arrival rate of customers to the system.

To use Little's Law for computing RT as seen by a particular DMU, identify the corresponding IN place, IP transition, and apply the following algorithm:

1. Determine the places that constitute the "system". This consists of all the places that belong to the same P-invariants as IN since a P-invariant corresponds to all the places through which a given token circulates.
2. Compute N as the total average number of tokens in these places as seen by IN.

3. Compute λ_e = dmu-rate prob{IP is enabled}, the effective arrival rate of tokens to the system.
4. Since PRO is an immediate transition, the time spent in OUT is zero.

   Therefore, RT = N/λ_e + 1/dmu-rate. The last term is needed since "system" does not take into account the time spent in IP.

As for other performance measures, the utilization or workload of the DMU is defined as the percentage of time it spends processing one or more requests. This is computed as

$$\text{DMU Utilization} = \text{prob}\{m(MP) \neq m(MP_{\text{initial}})\},$$

where $m(MP_{\text{initial}})$ represents the initial number of tokens in place MP. The average number of queued requests, and responses or commands is

$$\begin{aligned}\text{Queued Requests} &= E[m(IN)],\\ \text{Queued Responses} &= E[m(CMD)].\end{aligned}$$

7. PERFORMANCE EVALUATION OF STAR-1 DMA

The STAR-1 DMA in Figure 6 controls a physical process PRO using two computers configured in a two-level hierarchy. The GSPN model is shown in Figure 12. When PRO requests the intervention of the DMA, its request is processed either by DMU_{11} alone or by DMU_{11} and DMU_{21} as specified by hi-aut_{11}. If DMU_{11} can

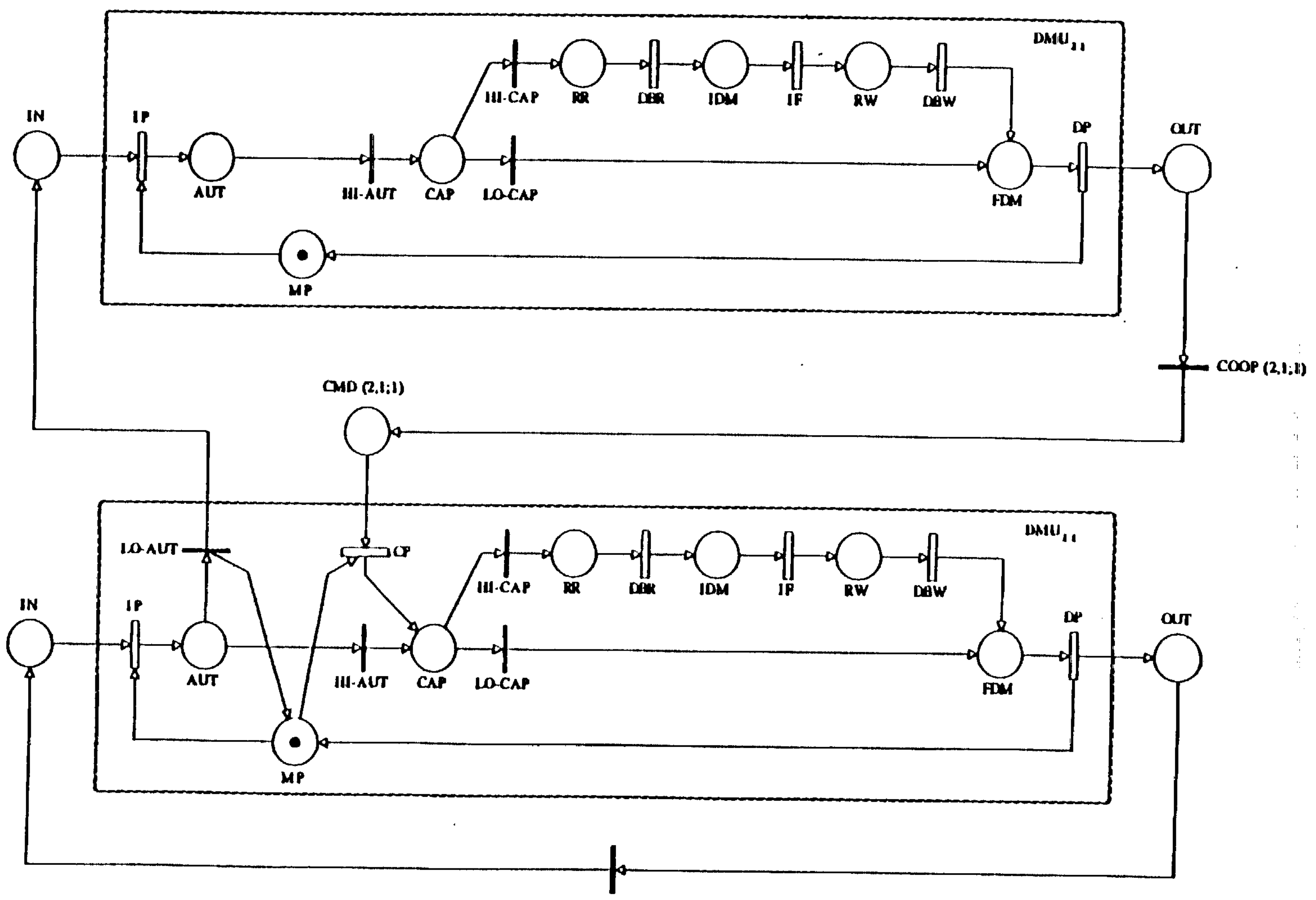


Figure 12: GSPN model of a STAR-1 DMA.

make the decision locally, then hi-cap$_{11}$ will determine the decision paths through DMU$_{11}$. If processing by DMU$_{21}$ is needed, then the decision paths through DMU$_{21}$ are determined by hi-cap$_{21}$. The STAR-1 DMA has six possible decision paths, representing the six basic T-invariants. Therefore, RT$_{11}$ is the average time a request spends while passing through all six possible decision paths.

STAR-1 has the following three basic P-invariants whose interpretations are similar to those in the previous section:

$$\begin{aligned}
&\sum [m(\text{all places}) \text{ except } \{m(MP_{11}), m(MP_{21})\}] = 1,\\
&m(AUT_{11}) + m(CAP_{11}) + m(RR_{11}) +\\
&\quad m(IDM_{11}) + m(RW_{11}) + m(FDM_{11}) + m(MP_{11}) = 1,\\
&m(AUT_{21}) + m(CAP_{21}) + m(RR_{21}) +\\
&\quad m(IDM_{21}) + m(RW_{21}) + m(FDM_{21}) + m(MP_{21}) = 1.
\end{aligned}$$

RT can be computed using the throughput method or Little's Law. The first P-invariant determines the set of places that constitute the "system" when applying Little's Law, as outlined in section 6. The average number of customers in the system as seen by IN$_{11}$ is

$$N_{11} = 1 - E[m(IN_{11})] \quad \text{tokens,}$$

and the effective arrival rate of tokens to the system is

$$\lambda_{e11} = \text{dmu-rate}_{11}\, \text{prob}\{m(IN_{11})\, m(MP_{11}) \neq 0\} \quad \text{tokens/time unit.}$$

Therefore, for this safe net, the response time is

$$\begin{aligned} RT_{11} &= N_{11}/\lambda_{e11} + 1/\text{dmu-rate}_{11} \\ &= 1/\text{dmu-rate}_{11}\, E[m(IN_{11})] \quad \text{time units} \end{aligned} \qquad (5),$$

which is the same as the throughput method. Intuitively, the places representing the multiprocessing capability of the DMUs should not contribute to the computation of RT. In Figure 12, there is only one request circulating in the system. Assigning MP more than one token would not affect the performance. Therefore, the number of tokens in MP could be made arbitrarily large. Were it to be included in the computation of N, the result would be an arbitrarily large RT, which is obviously wrong.

7.1 Response Time vs. Decision Making Rate

In this example, DMU_{11} responds on average to 50% of PRO's requests without requiring the intervention of DMU_{21}. Also on average, 50% of the requests to DMU_{11} require complex decision making algorithms and database accesses, similarly, for DMU_{21}. The effect of changing dmu-rate$_{11}$ on RT_{11} is shown in Figure 13. For dmu-rate$_{11} \leq 1$, RT_{11} is heavily influenced by dmu-rate$_{11}$, and the system behaves as an isolated DMU where DMU_{11} is the bottleneck. DMU_{21} is more dominant when dmu-rate$_{11} \geq 1$, and it becomes the bottleneck; there is virtually no improvement in RT_{11} even if DMU_{11} can make a decision in zero time. The overall response is inverse linearly proportional to dmu-rate$_{11}$ (linear in the decision making time of DMU_{11}) and the tail of the curve is a measure of the minimum achievable RT_{11} which represents the loading effects of DMU_{11} on DMU_{21}.

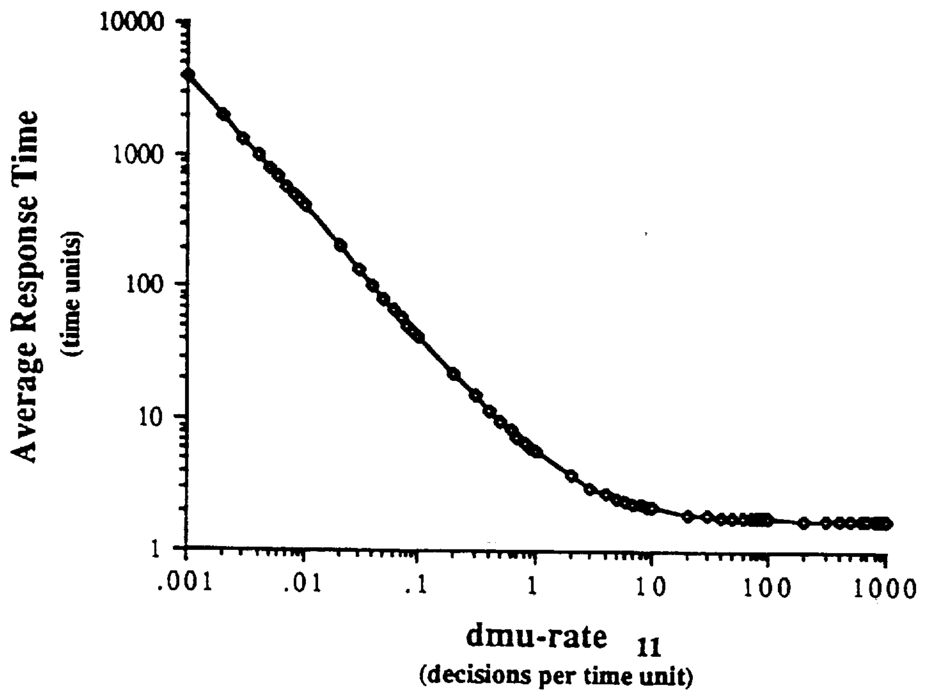


Figure 13: RT_{11} vs. dmu-rate$_{11}$ of a STAR-1 DMA.

7.2 Response Time vs. Degree of Autonomy

Figure 14 shows the effect of increasing the Degree of Autonomy of DMU_{11} on RT_{11} for three values of the Decision Making Capacity of DMU_{11}. In this example, the response time is more sensitive to changes in the Degree of Autonomy than in the Decision Making Capacity.

7.3 Markov Chain Analysis

The GSPN model for STAR-1 is simple enough that it allows for the derivation of a closed-form expression for RT_{11}. The Reachability Graph has 11 tangible and 6 vanishing markings. Proceeding as in section 5.2, Figure 15 shows the equivalent Markov Chain where $m(IN_{11}) = m(MP_{11}) = 1$ only in state 0, i.e. $\pi_0 = \text{prob}\{m(IN_{11})\, m(MP_{11}) \neq 0\}$. The balance equations are:

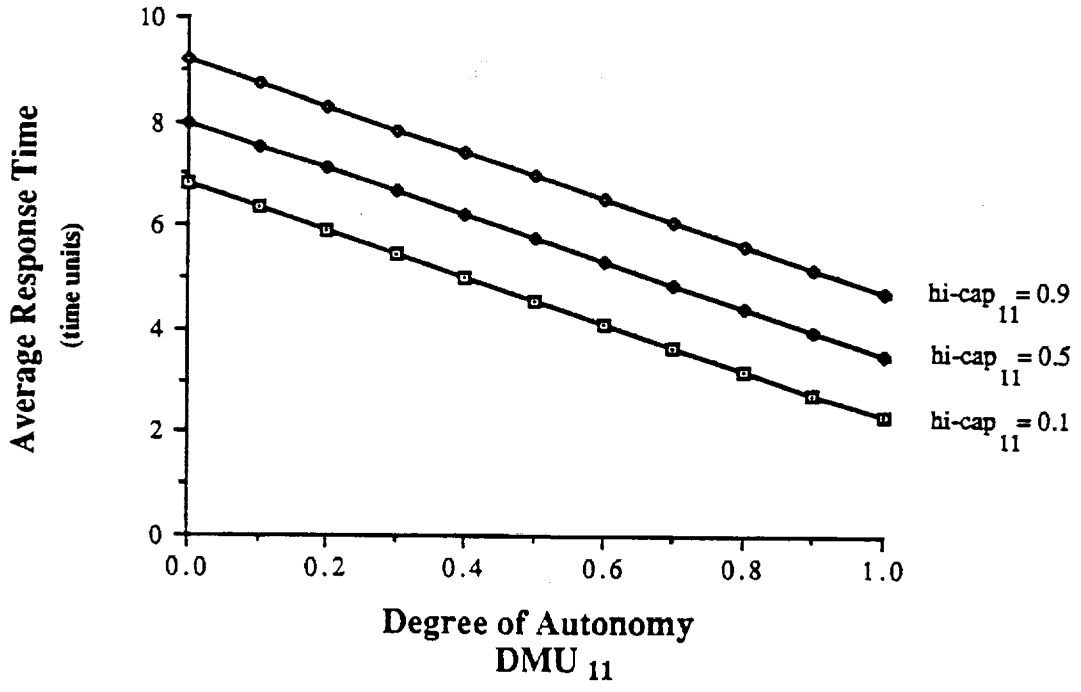


Figure 14: RT_{11} vs. hi-aut_{11} of a STAR-1 DMA.

$$\pi_1 = \pi_0,$$
$$\pi_2 = \pi_3 = (1 - \text{hi-aut}_{11})\,(\text{dmu-rate}_{11}/\text{dmu-rate}_{21})\,\pi_0,$$
$$\pi_4 = (1 - \text{hi-aut}_{11})\,\pi_0,$$
$$\pi_5 = \pi_6 = \pi_7 = \text{hi-cap}_{21}\,(1 - \text{hi-aut}_{11})\,(\text{dmu-rate}_{11}/\text{dmu-rate}_{21})\,\pi_0,$$
$$\pi_8 = \pi_9 = \pi_{10} = \text{hi-cap}_{11}\,\pi_0,$$

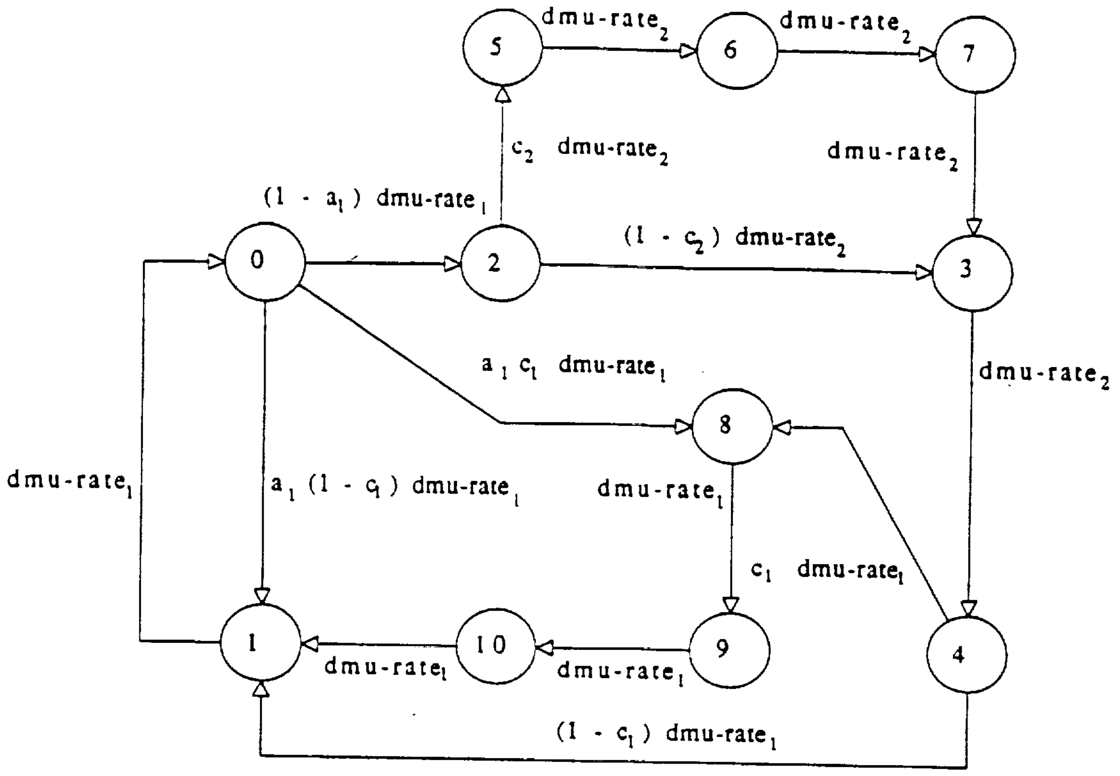


Figure 15: Equivalent Markov Chain for the STAR-1 DMA.

where the sum of all probabilities equals one. Therefore,

$$\pi_0 = 1/[2 + (1 + 2\rho + 3 c_2 \rho)(1 - a_1) + 3 c_1],$$

with

$$\begin{aligned} \rho &= \text{dmu-rate}_{11}/\text{dmu-rate}_{21} = r_1/r_2, \\ a_1 &= \text{hi-aut}_{11}, \\ c_1 &= \text{hi-cap}_{11}, \\ c_2 &= \text{hi-cap}_{21}. \end{aligned}$$

Then,

$$\begin{aligned} RT_{11} &= 1/\pi_0 \text{ dmu-rate}_{11} \\ &= [2 + (1 + 2\rho + 3 c_2 \rho)(1 - a_1) + 3 c_1]/\text{dmu-rate}_{11} && (6), \\ &= [2 + (1 - a_1) + 3 c_1]/\text{dmu-rate}_{11} + \\ &\quad (2 + 3 c_2)(1 - a_1)/\text{dmu-rate}_{21} \quad \text{time units} && (7). \end{aligned}$$

As a quick check, assume that all the times are deterministic with all values being standard, then RT_{11}, reflecting the average delay through all possible decision paths, can be computed as the weighted sum of all these paths. That is,

$$\begin{aligned} RT_{11} &= 2(1/4) + 5(1/4) + 5(1/8) + 8(1/8) + 8(1/8) + 11(1/8) \\ &= 5.75 \text{ time units,} \end{aligned}$$

in agreement with (6) when using the standard numerical values. Note that (5) and (6) are equal since $\pi_0 = \text{prob}\{m(IN_{11})\ m(MP_{11}) \neq 0\}$. Table 3 displays several expressions of RT_{11} for different values of a_1, c_1, and c_2.

	$c_2 = 1$	$c_2 = 0$
$a_1 = 0$	$3(1 + c_1)/r_1 + 5/r_2$	$3(1 + c_1)/r_1 + 2/r_2$
$a_1 = 1$	$(2 + 3\,c_1)/r_1$	$(2 + 3\,c_1)/r_1$
$c_1 = 1$	$(6 - a_1)/r_1 + 5(1 - a_1)/r_2$	$(6 - a_1)/r_1 + 2(1 - a_1)/r_2$
$c_1 = 0$	$(3 - a_1)/r_1 + 5(1 - a_1)/r_2$	$(3 - a_1)/r_1 + 2(1 - a_1)/r_2$

Table 3: Several expressions for RT_{11} of the STAR-1 DMA.

For $a_1 = 1$, the response time equals that of the isolated DMU. It is the shortest when $a_1 = 1$, and $c_1 = 0$, while for $c_2 = c_1 = 1$, and $a_1 = 0$, it is the longest.

The maximum increases in RT_{11} as a_1 and c_1 each change from 0 to 1 are

$$\Delta RT(a_1) = 1/r_1 + 5/r_2, \qquad \Delta RT(c_1) = 3/r_1, \qquad \text{for } c_2 = 1 \qquad (8),$$
$$\Delta RT(a_1) = 1/r_1 + 2/r_2, \qquad \Delta RT(c_1) = 3/r_1, \qquad \text{for } c_2 = 0 \qquad (9).$$

With $c_2 = 1$, $\Delta RT(a_1)$ in (8) reflects the addition of a decision path, consisting of one timed transition in DMU_{11} and five timed transitions in DMU_{21} (Figure 12), as DMU_{11} varies from complete autonomy to complete dependence on DMU_{21}. $\Delta RT(c_1)$ reflects the new path in DMU_{11}, consisting of three timed transitions, as its Decision Making Capacity increases. Similar observations can be made regarding (9). Note how $\Delta RT(c_1)$ is the same regardless of c_2. This is expected since the Decision Making Capacity c_1 is a property internal to DMU_{11}. From (8), changing hi-aut$_{11}$ will have more impact on RT_{11} than changing hi-cap$_{11}$ when

$$\text{dmu-rate}_{11} \geq (2/5)\ \text{dmu-rate}_{21}.$$

Similarly, from (9), this occurs when

$$\text{dmu-rate}_{11} \geq \text{dmu-rate}_{21}.$$

The second inequality represents the case with the shorter response time. Referring to the GSPN model in Figure 12, these inequalities show that processing a request through DMU_{21} could be less costly than that exclusively processing it in DMU_{11} which will depend on the Decision Making Rates.

Substituting the standard numerical values of section 4, (7) becomes

$$RT_{11} = 7/4 + 4/\text{dmu-rate}_{11} \quad \text{time units,} \quad \text{and}$$
$$RT_{11} = 13/2 + (3/2)(2\ \text{hi-cap}_{11} - 3\ \text{hi-aut}_{11}) \quad \text{time units,}$$

which verify Figures 13-14, respectively. As shown in Figure 14, for this STAR-1 DMA with $\text{dmu-rate}_{11} = \text{dmu-rate}_{21} = 1$, RT_{11} can be more efficiently improved by increasing hi-aut_{11} rather than the hi-cap_{11} of DMU_{11}, regardless of hi-cap_{21} of DMU_{21}. However, choosing $\text{dmu-rate}_{11} = 0.01$ results in the opposite behavior.

The above analysis shows that RT_{11} is inversely proportional to the Decision Making Rates, and linearly proportional to their Degrees of Autonomy and Decision Making Capacities. (Recall that, on a log-log scale, $y = 1/x$ is a monotonically decreasing straight line as in Figure 7, and $y = 1/x + k$ is a monotonically decreasing curve with a minimum of k as in Figure 13.) The first term of (7) is identical to (3) of the isolated DMU where $a_1 = 1$. The second term is a constant determined by the other numerical values as dmu-rate_{11} varies, and it represents the minimum value that RT_{11} can assume. In Figure 13, two operating regions can therefore be identified:

$$RT_{11} \approx (3 - a_1 + 3\,c_1)/\text{dmu-rate}_{11} \quad \text{time units} \quad \text{for } \rho \ll 1 \quad (10),$$
$$RT_{11} \approx (2 + 3\,c_2)(1 - a_1)/\text{dmu-rate}_{21} \quad \text{time units} \quad \text{for } \rho \gg 1 \quad (11).$$

Equation (10) shows that for $\rho \ll 1$, RT_{11} is more sensitive to changes in c_1 than in a_1 since c_1 is multiplied by 3. On the other hand, (11) shows that for $\rho \gg 1$, RT_{11} is not dependent on c_1. The choice of standard numerical values results in $\rho = 1$.

This explains why the response time in Figure 14 is more sensitive to changes in the Degree of Autonomy than in the Decision Making Capacity.

8. PERFORMANCE EVALUATION OF STAR-3 DMA

The STAR-3 DMA in Figure 6 controls a manufacturing system composed of three physical processes $\{PRO_1, PRO_2, PRO_3\}$ using four computers arranged in a two-level hierarchy. This could represent three workstations configured in one cell, with each workstation controlling several machines. Each process may request the intervention of the DMA as the need arises, which is then processed by at least one of the DMUs. The decision processing paths through the DMA are specified by the discrete probability distributions of the random switches.

8.1 Modeling Cooperation Between Decision Making Units

STAR-3 has a random switch COOP connected to OUT_{21} as can be seen in Figure 16. As stated in section 3.1, this is needed to direct the response from DMU_{21} to one of its lower-level DMUs. Unless explicitly stated otherwise, all the immediate transitions of COOP will be assigned equal probabilities.

Because of the random nature of COOP, a response from DMU_{21} will not necessarily be sent to the low-level DMU that actually issued the request. RT reflects the average time it takes a request to flow through <u>all</u> the possible decision paths. This implies that a request by PRO_1, for example, will be processed by <u>all</u> the DMUs, as

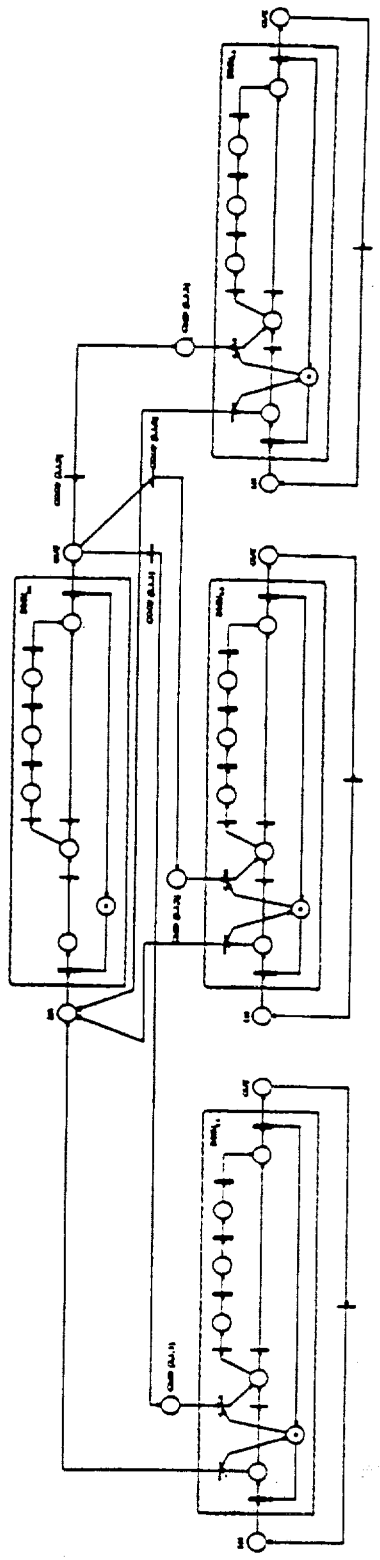


Figure 16: GSPN model of a RANDOM STAR-3 DMA.

specified by the random switches in the model. Hence, RT *measures the average time that elapses between sending a request for intervention and receiving a response, not necessarily to the same request.* This apparent ambiguity is due to the fact that all the requests are indistinguishable, so are all the responses. Such models will be called RANDOM to reflect the nature of selecting the destination DMU as opposed to First-In First-Out (FIFO) models discussed later. Section 3.1 presented an example on the use and interpretation of COOP. If it is desired to model a situation where, say, PRO_1's request is processed by DMU_{11} and if needed, by DMU_{12} only, then a FIFO model should be used. In this case, DMU_{12} and DMU_{13} do not participate in making a decision, although their loading effect is reflected by the increase in the average number of requests that DMU_{21} processes. Here, RT *measures the average time that elapses between sending a request for intervention and receiving the response to that same request.* The proposed methodology, along with the algorithm described in the Appendix, enables the development of the desired live, bounded, and reversible FIFO models. This will be demonstrated in section 8.3. Both models and corresponding measures are correct. However, they have two different interpretations, and it is the responsibility of the modeler to choose the appropriate model that best describes the physical system at hand.

8.2 RANDOM STAR-3 DMA

The Reachability Graph of the GSPN model in Figure 16 has 3380 markings, while the Embedded Markov Chain has 1460 states. Since the GSPN model is not safe, Little's Law will be used to compute the response time. There are five basic P-invariants:

Σ[m(all places) except {m(MP_{11}), m(MP_{12}), m(MP_{13}), m(MP_{21})}] = 3,
m(AUT_{11}) + m(CAP_{11}) + m(RR_{11}) +
m(IDM_{11}) + m(RW_{11}) + m(FDM_{11}) + m(MP_{11}) = 1,
m(AUT_{12}) + m(CAP_{12}) + m(RR_{12}) +
m(IDM_{12}) + m(RW_{12}) + m(FDM_{12}) + m(MP_{12}) = 1,
m(AUT_{13}) + m(CAP_{13}) + m(RR_{13}) +
m(IDM_{13}) + m(RW_{13}) + m(FDM_{13}) + m(MP_{13}) = 1,
m(AUT_{21}) + m(CAP_{21}) + m(RR_{21}) +
m(IDM_{21}) + m(RW_{21}) + m(FDM_{21}) + m(MP_{21}) = 1.

Using the first P-invariant

N_{11} = 3 - E[m(IN_{11})] tokens,
N_{12} = 3 - E[m(IN_{12})] tokens,
N_{13} = 3 - E[m(IN_{13})] tokens,

and the effective arrival rates of tokens to each of these "systems" are

λ_{e11} = dmu-rate$_{11}$ prob{m(IN_{11}) m(MP_{11}) $\neq$ 0} tokens/time unit,
λ_{e12} = dmu-rate$_{12}$ prob{m(IN_{12}) m(MP_{12}) $\neq$ 0} tokens/time unit,
λ_{e13} = dmu-rate$_{13}$ prob{m(IN_{13}) m(MP_{13}) $\neq$ 0} tokens/time unit.

Therefore, the response times are

RT_{11} = N_{11}/λ_{e11} + 1/dmu-rate$_{11}$ time units,
RT_{12} = N_{12}/λ_{e12} + 1/dmu-rate$_{12}$ time units,
RT_{13} = N_{13}/λ_{e13} + 1/dmu-rate$_{13}$ time units.

8.2.1 Response Time vs. Decision Making Rate

Figure 17 shows the effects of varying dmu-rate$_{11}$ on RT$_{11}$, RT$_{12}$ and RT$_{13}$. The chosen numerical values result in RT$_{12}$ = RT$_{13}$. Guided by the STAR-1 analysis, this response suggests an inverse linear dependency on dmu-rate$_{11}$ with the loading effects being slightly higher on RT$_{11}$ than on RT$_{12}$ and RT$_{13}$. Two operating regions can also be identified. As dmu-rate$_{11}$ increases, all response times decrease substantially until dmu-rate$_{11}$ equals 1. From this point on, any increase in dmu-rate$_{11}$ will have a negligible effect on the response times, and DMU$_{11}$ ceases to be the dominant unit in the DMA. The overall response time will then be dominated by one or more of the other DMUs. The overlap between the two curves occurs at dmu-rate$_{11}$ = 1 since it represents a symmetric point for the chosen standard numerical values. As dmu-rate$_{11}$ increases, the prob{m(IN$_{11}$) $\geq$ 1} decreases while prob{m(IN$_{12}$) $\geq$ 1} and prob{m(IN$_{13}$) $\geq$ 1} increase. Also, N$_{11}$ becomes larger than N$_{12}$ and N$_{13}$. In other words, more tokens spend more time in DMU$_{12}$ and DMU$_{13}$ than in DMU$_{11}$. This means that the effects of increasing the speed of DMU$_{11}$ are more noticeable for DMU$_{12}$ and DMU$_{13}$ which is an interesting result. This occurs because DMU$_{11}$ has a direct path from place IN to place OUT through the transition HI-AUT. Similar arguments can be made when dmu-rate$_{12}$ = dmu-rate$_{13}$ = 1000 and dmu-rate$_{12}$ = dmu-rate$_{13}$ = 0.001 as shown in Figures 18-19, respectively.

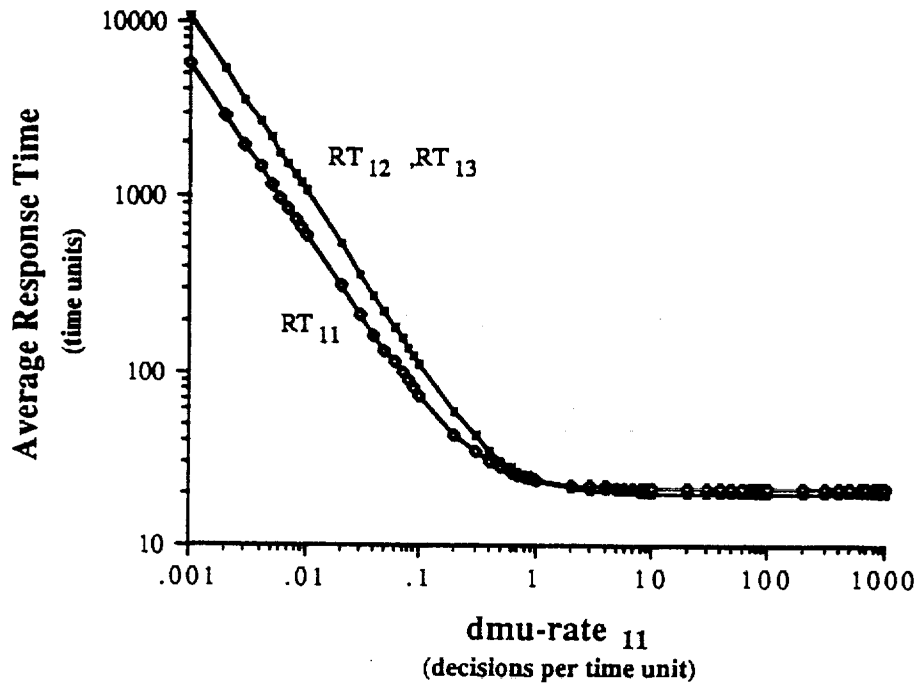


Figure 17: Response times vs. dmu-rate$_{11}$ of a RANDOM STAR-3 DMA.

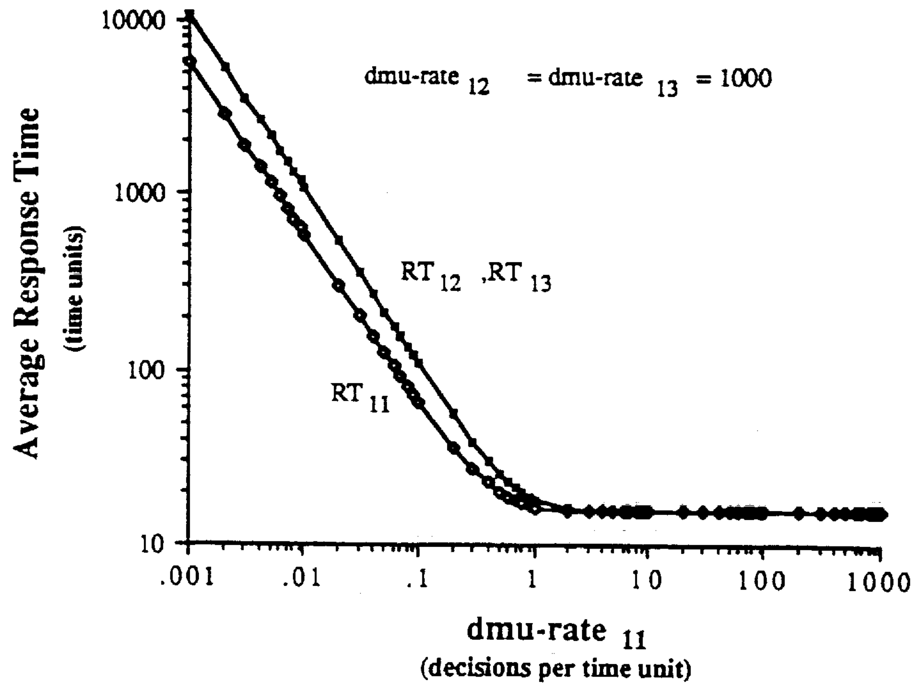


Figure 18: Same as Figure 17 with dmu-rate$_{12}$ = dmu-rate$_{13}$ = 1000.

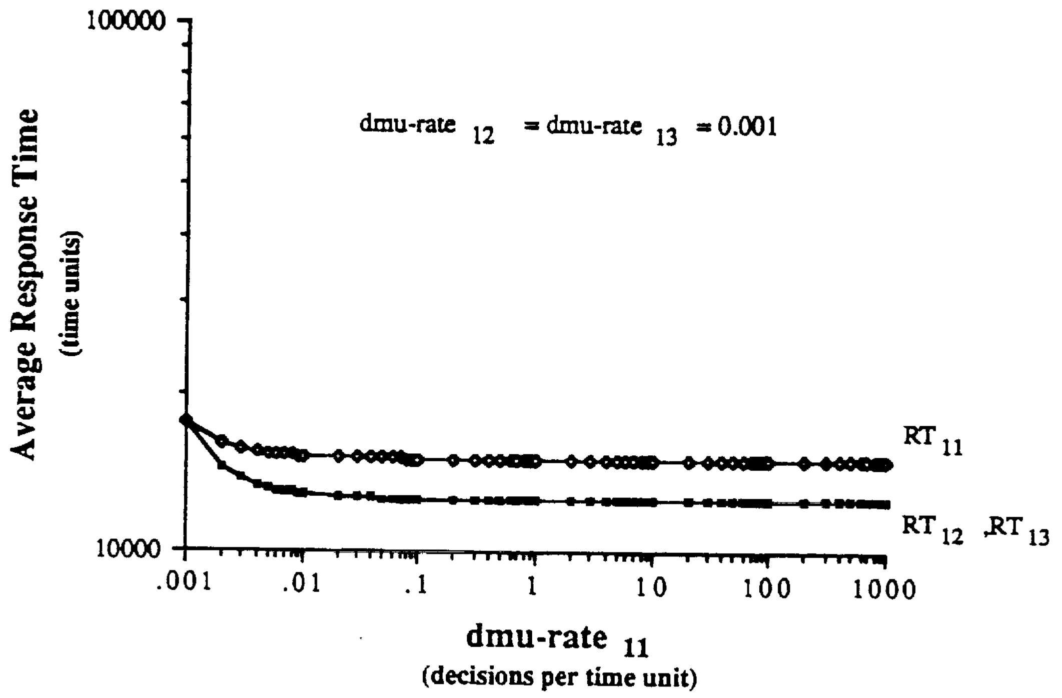


Figure 19: Same as Figure 17 with dmu-rate$_{12}$ = dmu-rate$_{13}$ = 0.001.

The topological structure of the Embedded Markov Chains in the above three cases are identical; they have the same behavior since the only differences are the transition rates from some of the states. Notice how Figure 19 is identical to the region where dmu-rate$_{11}$ ≥ 1 in Figure 17.

All of the above cases assumed that the DMUs had no multiprocessing capability and so m(MP) = 1. One would expect that with m(MP) = 3 the response times would be smaller. (The GSPN model for this case has a Reachability Graph with 5612 markings, and an Embedded Markov Chain with 2300 states.) Figure 20 verifies this and shows that all the response times are almost equal, which was not the case in Figure 17, but they still exhibit a similar relationship with dmu-rate$_{11}$. Since DMU$_{11}$ is very slow compared to the other DMUs, it benefited the least from the addition of this multiprocessing capability when dmu-rate$_{11} \leq 1$. However, for dmu-rate$_{11} \geq 1$, all the response times were reduced almost equally.

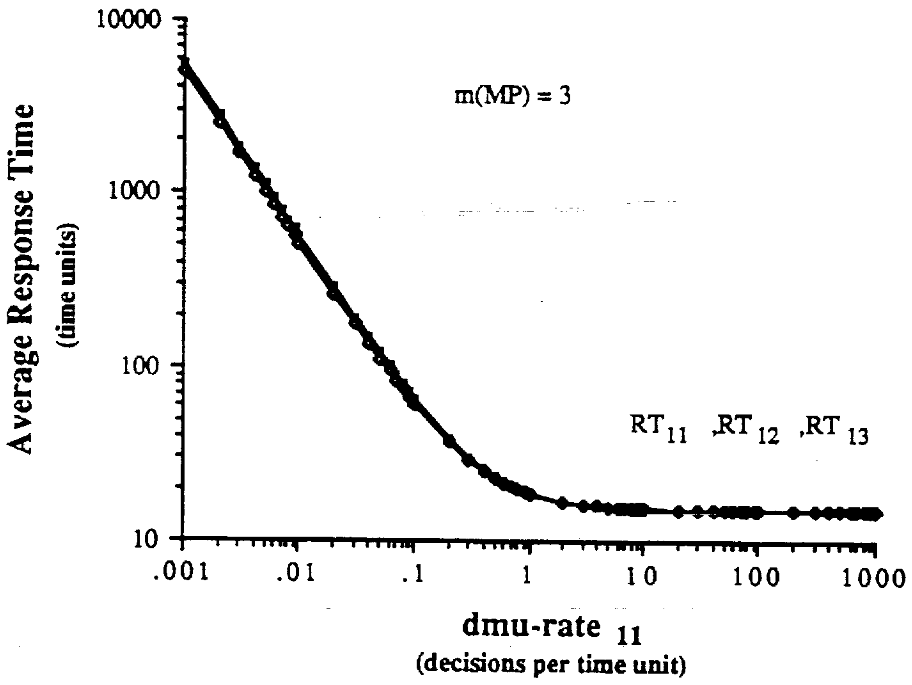


Figure 20: Same as Figure 17 with m(MP) = 3.

8.2.2 Response Time vs. Degree of Autonomy

Figure 21 shows the effect of increasing the Degree of Autonomy, hi-aut$_{11}$, for three values of Decision Making Capacity, hi-cap$_{11}$. Changing hi-aut$_{11}$ has more impact on the response times than changing hi-cap$_{11}$. As explained in section 7.3, this is due to the ratio of the DMU rates. In this example, if one is interested in reducing the response time of the DMA, then it is more beneficial to give each DMU the capability to make more decisions locally rather than requesting the intervention of a higher-level DMU. Obviously, this will not be the case for all possible situations. Still, this important result confirms a rule of thumb in the design of hierarchical control systems. Johnson points out that "in a well-balanced hierarchical control system, the tasks at the same level should share the same degree of complexity, similar completion times, comparable degrees of uncertainty..." and that "errors should be identified and resolved at the lowest level possible..." [2]. The RANDOM STAR-3 DMA represents a well-balanced system. The standard numerical values reflect equal DMU rates, and comparable degrees of autonomy and capacity. Giving the lowest-level DMUs more autonomy implies that they are more responsible for responding to requests from the manufacturing process. Hence, the above analysis gives the designer valuable insight that could aid in improving the performance.

The above analysis shows that RT_{11} is linearly dependent on hi-aut_{11}. The same linear relationship between RT_{11} and hi-aut_{11} and hi-cap_{11}, found in the STAR-1 DMA, holds for this model, too.

As the Degree of Autonomy, hi-aut_{11}, increases, one expects RT_{12} and RT_{13} to decrease since the number of requests sent by DMU_{11} to DMU_{21} decreases. This does not occur because a response by DMU_{21} is directed to the three lower-level DMUs in a RANDOM, not FIFO order, as defined by the random switch COOP. As hi-aut_{11} increases there will be more tokens circulating within DMU_{11}. When DMU_{11} is completely autonomous the three tokens in the system never leave DMU_{11}. Hence, no token ever returns to DMU_{12} or DMU_{13} resulting in an infinite response time as shown

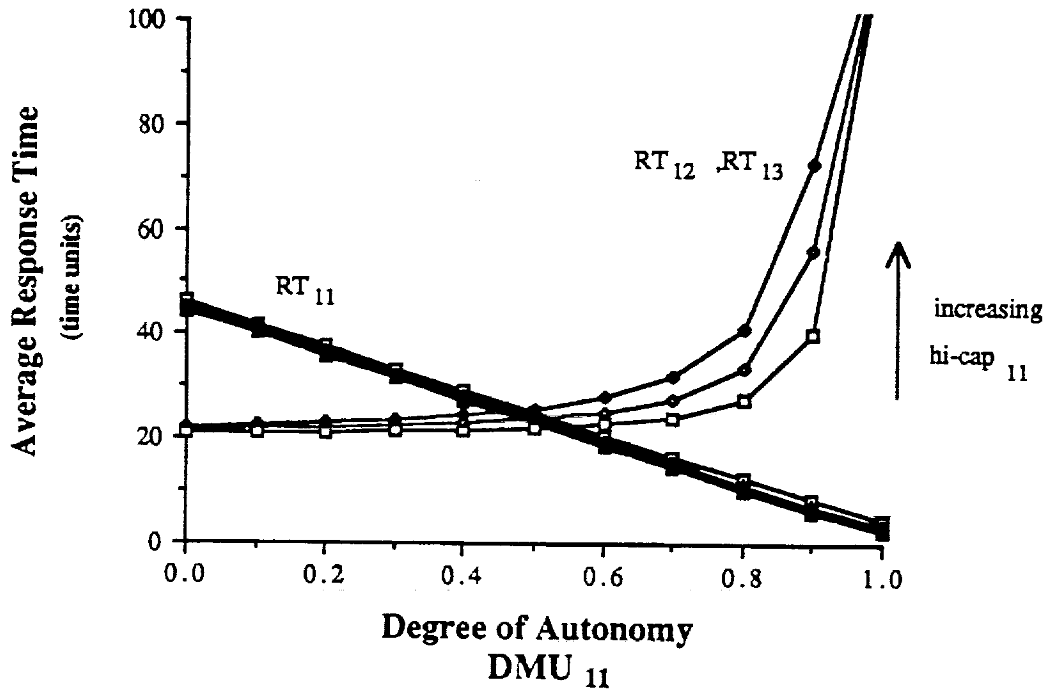


Figure 21: Response times vs. hi-aut$_{11}$ of a RANDOM STAR-3 DMA.

in Figure 21.

Two final observations. First, increasing hi-cap$_{11}$ increases the response times. Second, RT$_{11}$, RT$_{12}$ and RT$_{13}$ intersect at the point where hi-aut$_{11}$ = 0.5 which indicates a completely symmetric model for the chosen standard numerical values.

8.2.3 Response Time vs. Degree of Cooperation

Two choices for the discrete probability distribution of COOP will be compared. They are (0.9, 0.05, 0.05) and (0.1, 0.45, 0.45). The first choice means that, on average, 90% of the decisions processed by DMU_{21} will be directed to DMU_{11}. It models a situation where two of the three workstations under a cell controller are heavily dependent on sharing a resource, such as a robot, an AGV, or database information, that the first one controls. The second choice directs, on average, only 10% of these decisions to DMU_{11}. The remaining decisions are equally distributed between the other two DMUs. This model could represent three workstations controlled by a cell, where two workstations are heavily-cooperating. Figure 22 compares these two choices as they affect RT_{11}. The first choice results in a lower response time since more tokens will be sent more often to DMU_{11}. Figure 23 shows the effect of the two probability choices on RT_{12} and RT_{13}, where $RT_{12} = RT_{13}$. The first choice leads

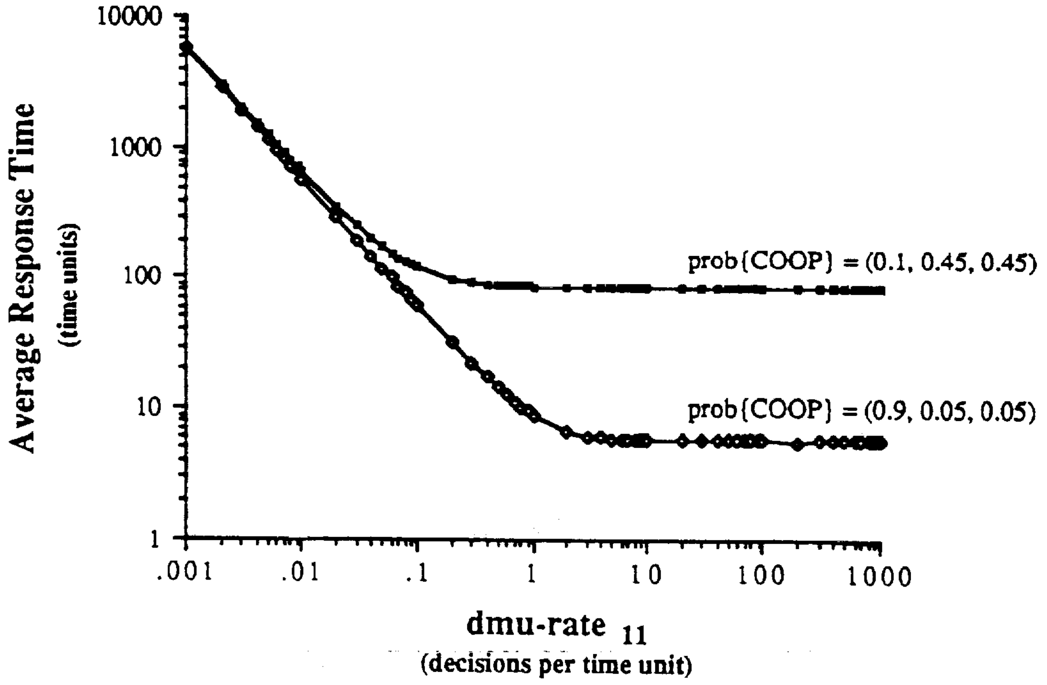


Figure 22: RT_{11} vs. dmu-rate$_{11}$ of a RANDOM STAR-3 DMA.

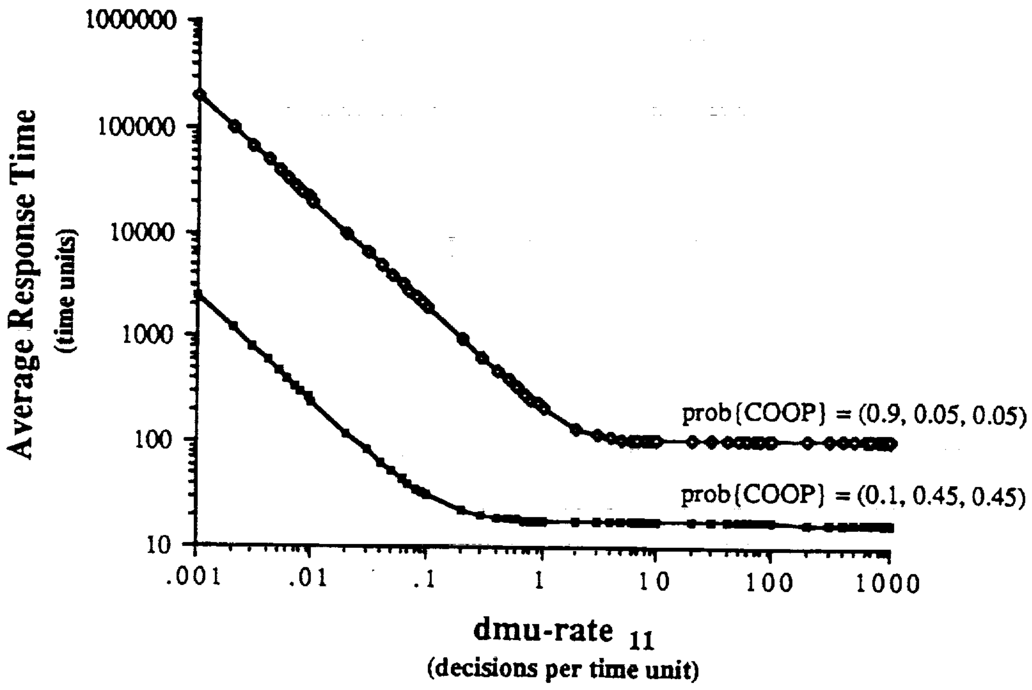


Figure 23: RT_{12} and RT_{13} vs. dmu-$rate_{11}$ of a RANDOM STAR-3 DMA.

to much larger response times since the tokens will be directed to DMU_{12} or DMU_{13} only 5% of the time. They have to wait for a much longer time to receive a response.

8.3 FIFO STAR-3 DMA

In a FIFO scenario, a request for intervention is processed only by the corresponding lower-level DMU and its parent DMU_{21}. There is no need for the random switch COOP, and instead, the model in Figure 16 must be modified to capture the desired behavior. Ideally, the desired FIFO behavior is elegantly modeled using FIFO nets [31] or Colored GSPNs [32].

Figure 24 shows a simple FIFO net where three workstations t_a, t_b and t_c share a pool of robots (not shown) by queueing their requests in a buffer p_Q. The marking of the place p_Q is a string whose symbols represent the order in which the requests arrived. If a robot becomes available, it is allocated to the workstation that requested it first. When a workstation issues a request to p_Q (t_a, t_b or t_c fires), the workstation's label, which is attached to the corresponding arc will be added as a postfix to the string representing the current marking. Similarly, when a workstation acquires a robot (t_A, t_B or t_C fires), it removes the first symbol of the marking string. Note that a workstation can acquire a robot only if the first symbol of the marking string is identical to the symbol attached to its corresponding arc. To demonstrate this, let the marking string be acb which indicates that workstation A, then C, then B requested a robot. According to the firing rules described above, workstation A will get the first available robot, then workstation C, and finally workstation B.

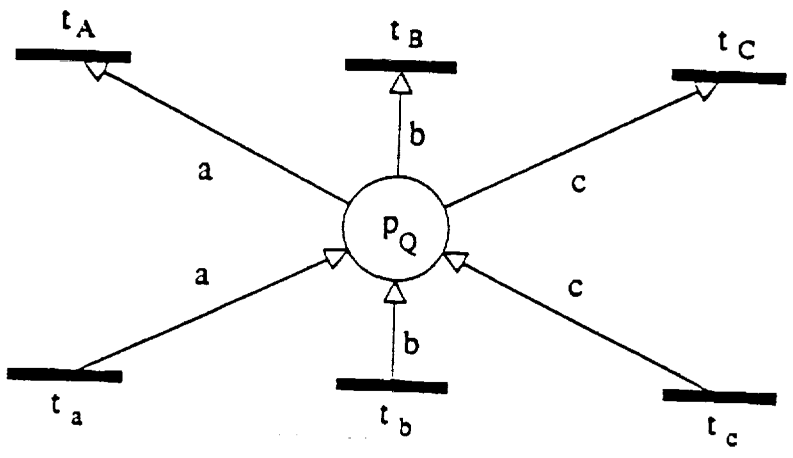


Figure 24: A simple example of a FIFO net.

Due to the lack of software that models FIFO nets or Colored GSPNs, some control logic is needed to emulate the desired FIFO behavior using GSPNs, and concepts from Adaptive PNs [33], and implementations of marking-dependent arcs [34]. Figure 25 shows one possible implementation of such logic used with the model in Figure 16. The resulting GSPN model has a Reachability Graph with 2370 markings, and an Embedded Markov Chain with 966 states. Details of the algorithm needed to synthesize this logic can be found in the Appendix.

Since only one token circulates in each of the lower level DMUs, the response times are computed using the throughput method where

$$\begin{aligned} RT_{11} &= 1/\text{dmu-rate}_{11}\ \text{prob}\{m(IN_{11})\ m(MP_{11}) \neq 0\} && \text{time units,} \\ RT_{12} &= 1/\text{dmu-rate}_{12}\ \text{prob}\{m(IN_{12})\ m(MP_{12}) \neq 0\} && \text{time units,} \\ RT_{13} &= 1/\text{dmu-rate}_{13}\ \text{prob}\{m(IN_{13})\ m(MP_{13}) \neq 0\} && \text{time units.} \end{aligned}$$

Here, RT_{1i} reflects the decision paths between DMU_{1i} and DMU_{21} rather than of all decision paths in the DMA as with the RANDOM models.

8.3.1 Response Time vs. Decision Making Rate

Figure 26 shows the effects of varying the Decision Making Rate, dmu-rate_{11}, on RT_{11}, RT_{12} and RT_{13}. The chosen numerical values result again in $RT_{12} = RT_{13}$. RT_{11} has a shape similar to that of the RANDOM model, indicating a similar linear relationship as described earlier, but with smaller values. However, RT_{12}

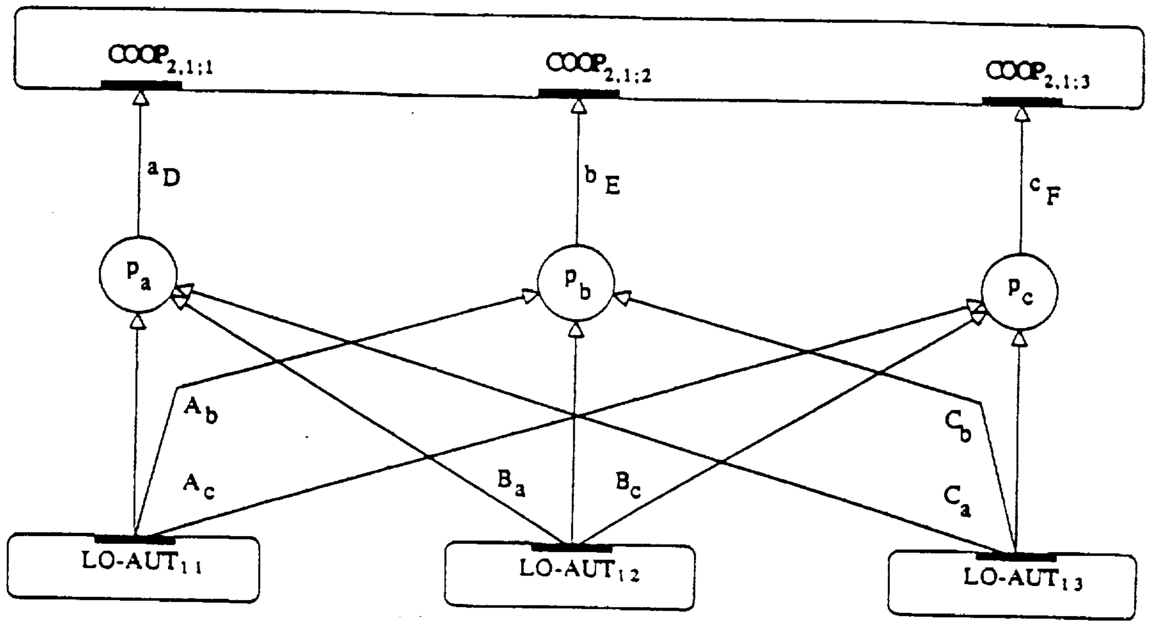


Figure 25: Control logic for a FIFO STAR-3 DMA.

and RT_{13} are almost insensitive to increases in the speed of DMU_{11}. To understand this behavior, note that such increases result in more requests sent to DMU_{21}, which will tend to increase the response time of DMU_{12} and DMU_{13}, albeit slightly. Note that increasing the speed of DMU_{11} beyond the crossover point will cause a significant improvement in RT_{11} while slightly increasing RT_{12} and RT_{13}. One can then conclude that the DMUs in the FIFO DMA are more decentralized. This is expected since the decision paths are only vertical. Figures 27-28 compare the RANDOM and FIFO cases for RT_{11}, RT_{12} and RT_{13}, respectively, where $RT_{12} = RT_{13}$.

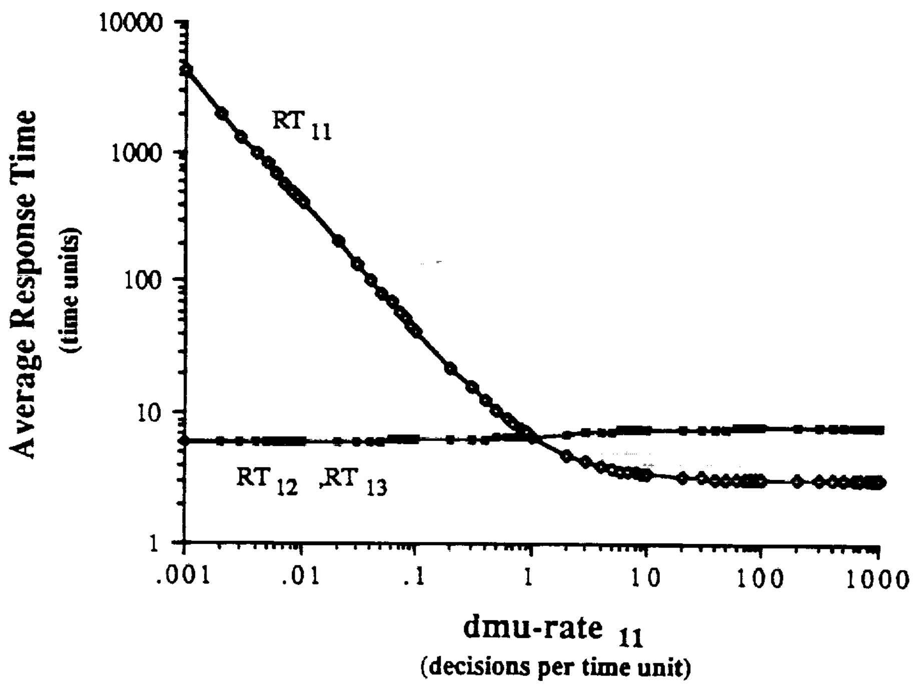


Figure 26: Response times vs. dmu-rate$_{11}$ of a FIFO STAR-3 DMA.

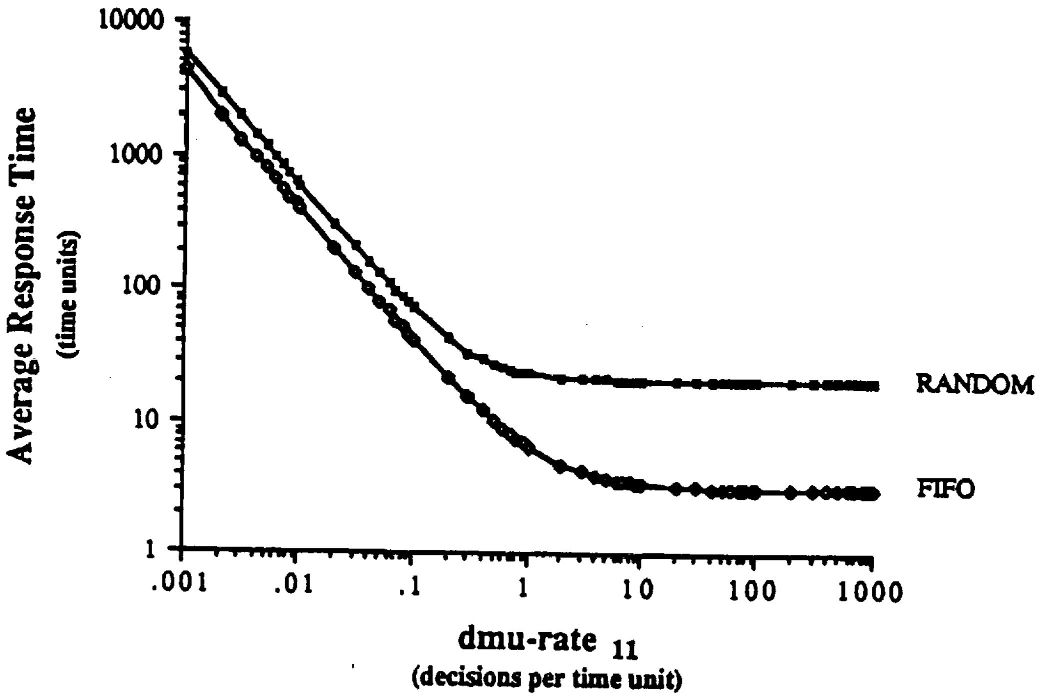


Figure 27: RT_{11} vs. dmu-rate$_{11}$ of a RANDOM and FIFO STAR-3 DMA.

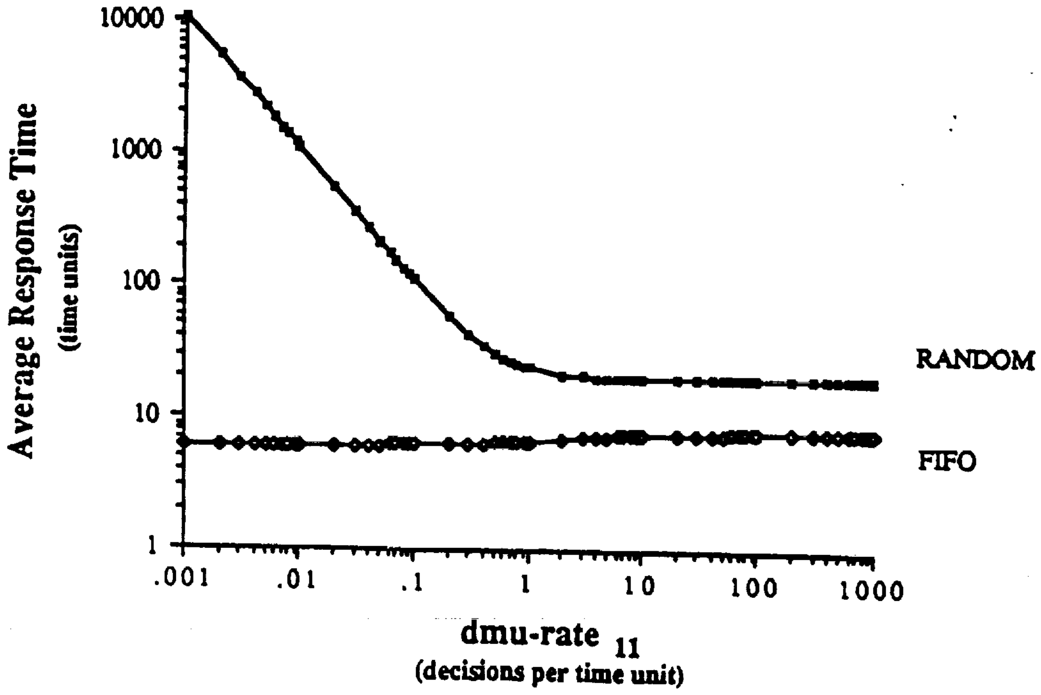


Figure 28: RT_{12} and RT_{13} vs. dmu-rate$_{11}$ of a RANDOM and FIFO STAR-3 DMA.

8.3.2 Response Time vs. Degree of Autonomy

Figure 29 shows the effects of changing the Degree of Autonomy, hi-aut$_{11}$, on the response times for three values of the Decision Making Capacity, hi-cap$_{11}$. As in the RANDOM case and for the standard numerical values, changing hi-aut$_{11}$ has more impact on the response times than changing hi-cap$_{11}$ and increasing hi-cap$_{11}$ results in larger response times. The decreasing linear relationship between RT$_{11}$ and hi-aut$_{11}$ is also preserved. The effect of increasing hi-aut$_{11}$ on RT$_{12}$ and RT$_{13}$ is negligible; they decrease only slightly due to the decrease in the average number of requests sent to DMU$_{21}$ by DMU$_{11}$. Note that RT$_{11}$, RT$_{12}$ and RT$_{13}$ intersect at different points depending on the value of hi-cap$_{11}$, while in the RANDOM case they intersected at the same point. This is attributed to the asymmetric nature of the FIFO model. When hi-aut$_{11}$ = 1, RT$_{12}$ and RT$_{13}$ are equal for all values of hi-cap$_{11}$. At this point DMU$_{11}$ is effectively isolated from the rest of the DMA and any changes in its parameters will have no effect on the rest of the DMA. Figure 30 compares the RANDOM and the FIFO cases.

From the results in Figures 27-28 and 30, it is apparent that the FIFO behavior, or a combination of FIFO and RANDOM behavior, is more realistic than a purely RANDOM one. The combination of FIFO and RANDOM models could represent a situation where a response is not always sent to the requester, but rather

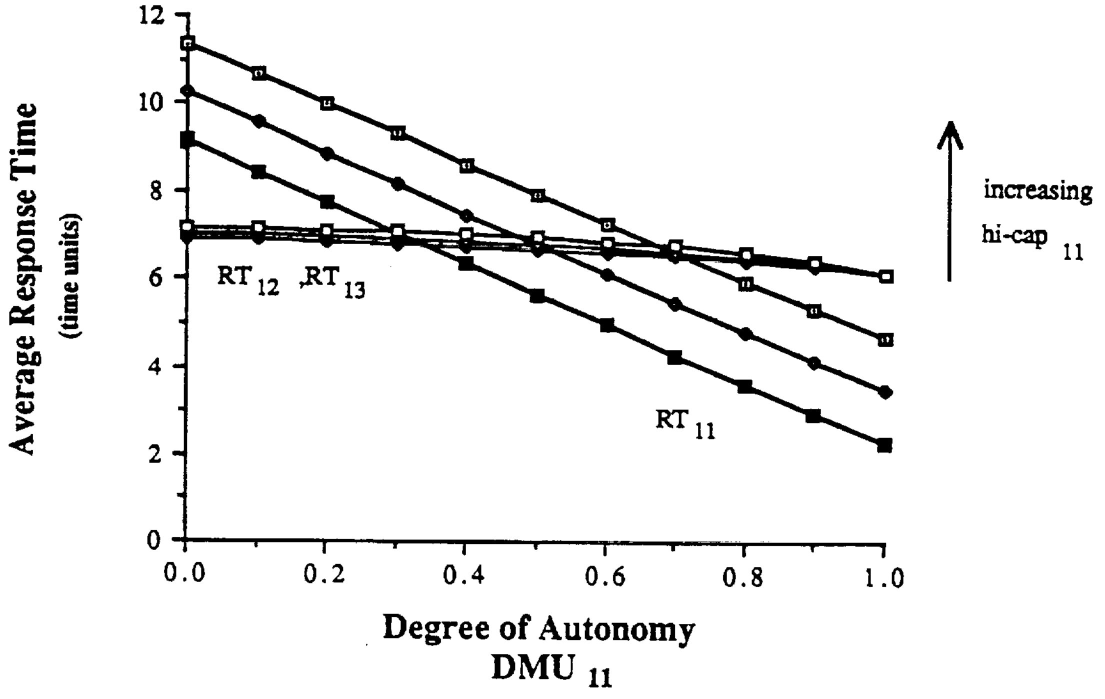


Figure 29: Response times vs. hi-aut_{11} of a FIFO STAR-3 DMA.

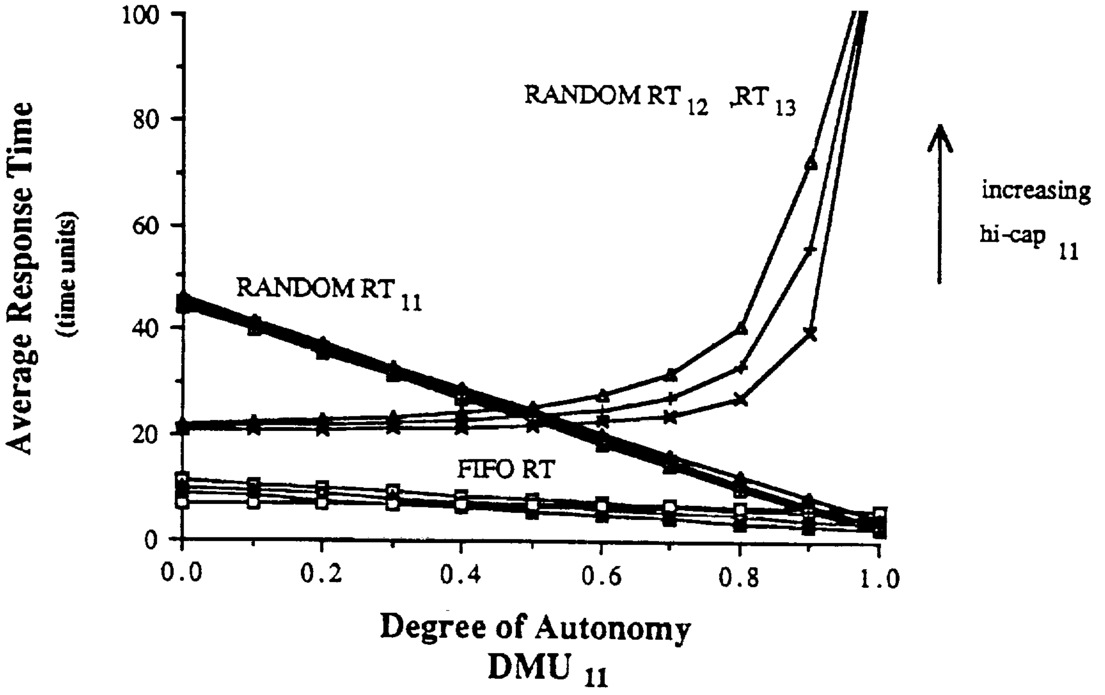


Figure 30: Response times vs. hi-aut_{11} of a RANDOM and FIFO STAR-3 DMA.

obeys some probability distribution. The difference from the RANDOM case is that the Degree of Cooperation is dependent on the DMU that issued the request. The proposed methodology is flexible enough that it allows for the construction of such mixed models. Contrary to all the models studied in this work, neither the DMUs nor the synthesis algorithm in the Appendix guarantee that the resulting model is live, bounded, and reversible; this is the responsibility of the modeler. It is best to model these situations using FIFO nets or Colored GSPNs.

8.4 RANDOM STAR-3 DMA with a Common Database

The DMU module has been designed with databases in mind although in this work database models and management protocols will not be extensively studied. The modularity of the proposed methodology allows for detailed database models to be incorporated by merely plugging them in place of the transitions DBR and DBW. The two cases discussed previously assume that each DMU has its own local database and that some database management system ensures that all the databases are consistent and promptly updated. The ease with which detailed database models can be incorporated and analyzed is demonstrated next.

Figure 31 shows the changes in the GSPN model of Figure 16 needed to control the access to a centralized database shared by the four DMUs. Database access is accorded on a First-Come First-Served basis. If two DMUs request access simultaneously, then on average the one with the shortest time for DBR or DBW wins the conflict. Any DMU can read as long as no other DMU is writing. All four DMUs can be simultaneously engaged in a DBR. (In fact, a maximum of three DBRs can

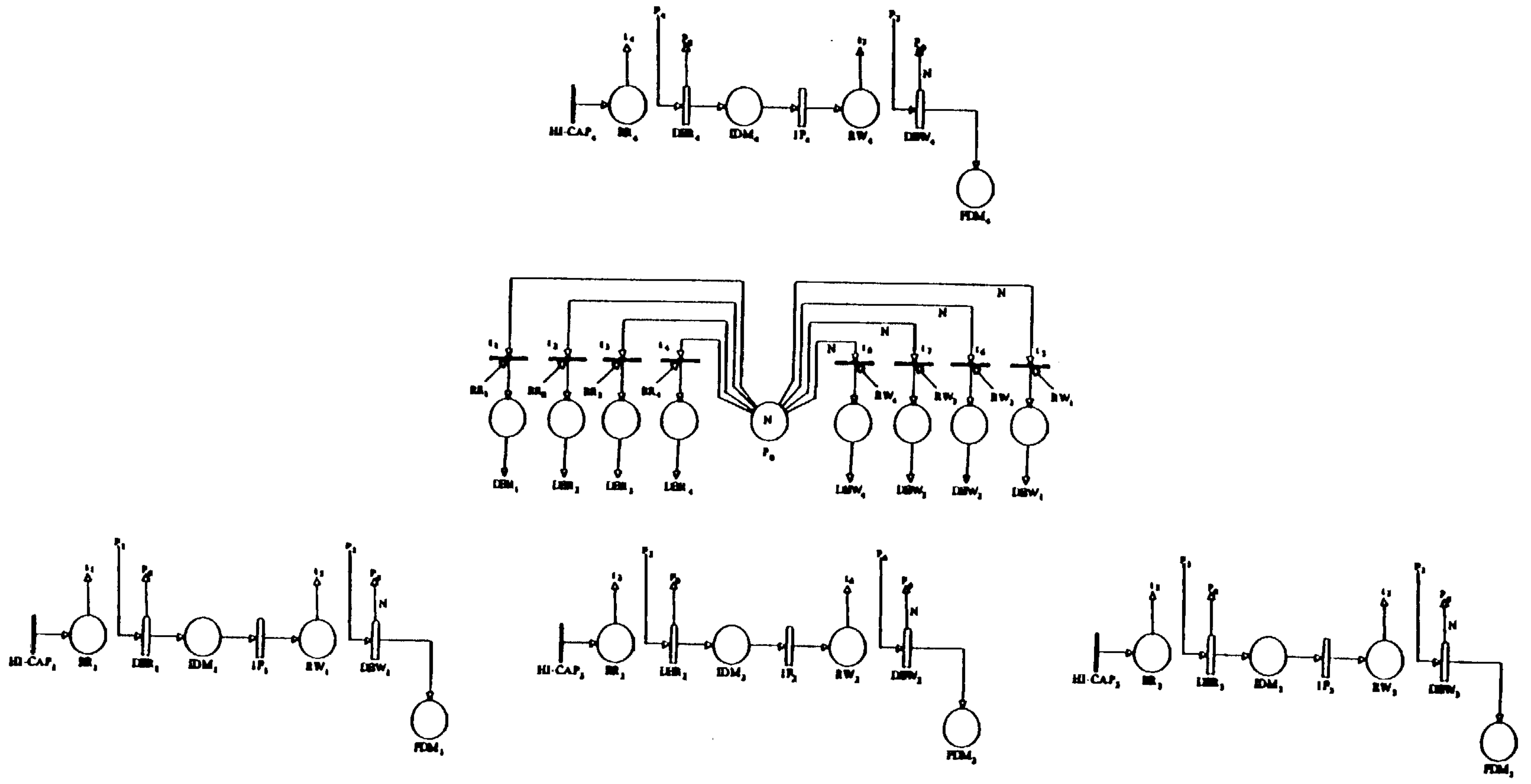


Figure 31: GSPN model of a RANDOM STAR-3 DMA with a common database.

occur simultaneously since there are only three requests in the system.) When any of the DMUs is writing, the remaining DMUs are locked out and can neither read nor write. The resulting model has a Reachability Graph with 4614 markings, and an Embedded Markov Chain with 1702 states.

The response times are expected to be larger than the RANDOM case. Figures 32-33 show the increase and percentage increase in RT_{11}, RT_{12} and RT_{13}. The existence of a peak is surprising. The peak percentage value for RT_{11} is 5.9% and 5.5% for RT_{12} and RT_{13}. As dmu-rate$_{11}$ becomes very large the percentage increase becomes steady at 4.5% for RT_{11} and 4.6% for RT_{12} and RT_{13}. The intersection of the

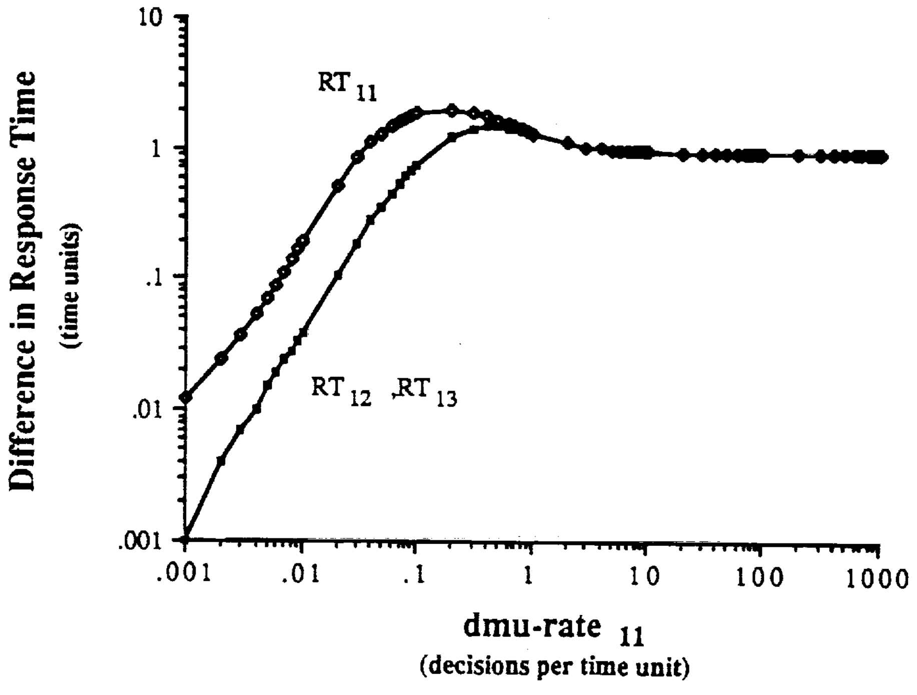


Figure 32: Increase in response times for a RANDOM STAR-3 DMA with a common database.

two curves at dmu-rate$_{11}$ = 1 is expected due to the symmetries in the model at that point. Figure 34 shows the percentage increase in the response times when hi-cap = 0.9, which implies a heavier use of databases and larger values for RT. Here, the peak value for RT$_{11}$ is 11.6% and 10.8% for RT$_{12}$ and RT$_{13}$. As dmu-rate$_{11}$ becomes very large the percentage increase becomes steady at 8.6% for RT$_{11}$ and 8.8% for RT$_{12}$ and RT$_{13}$. The above results indicate that the percentage increase in the response time behaves linearly as hi-cap$_{11}$ changes. Although to fully explain these curves requires further investigation, it is conjectured that the common database introduces strong coupling among the DMUs, which results in this behavior.

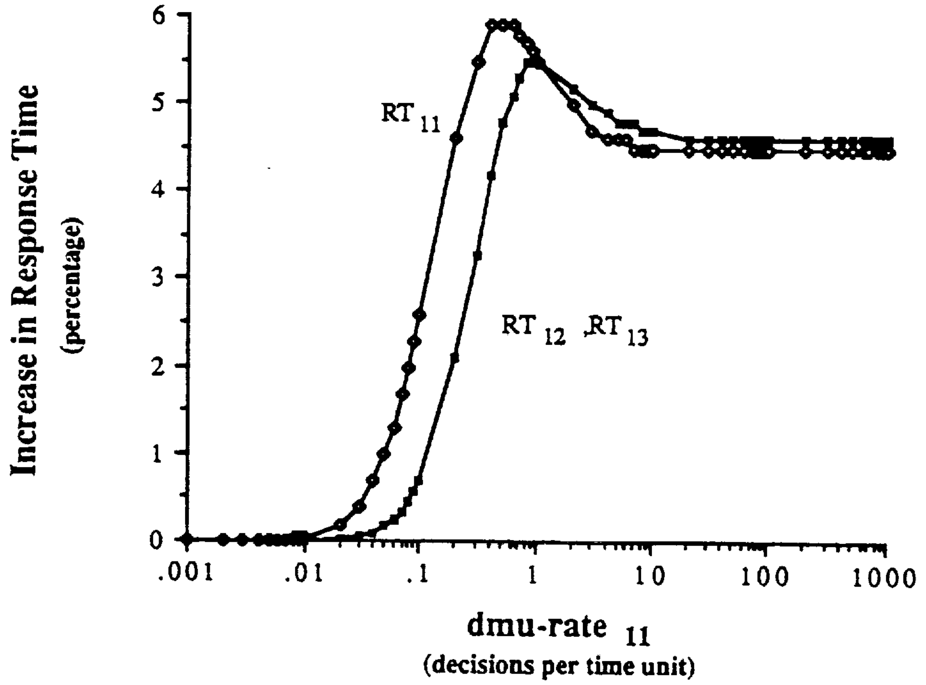


Figure 33: Percentage increase in response times of Figure 32.

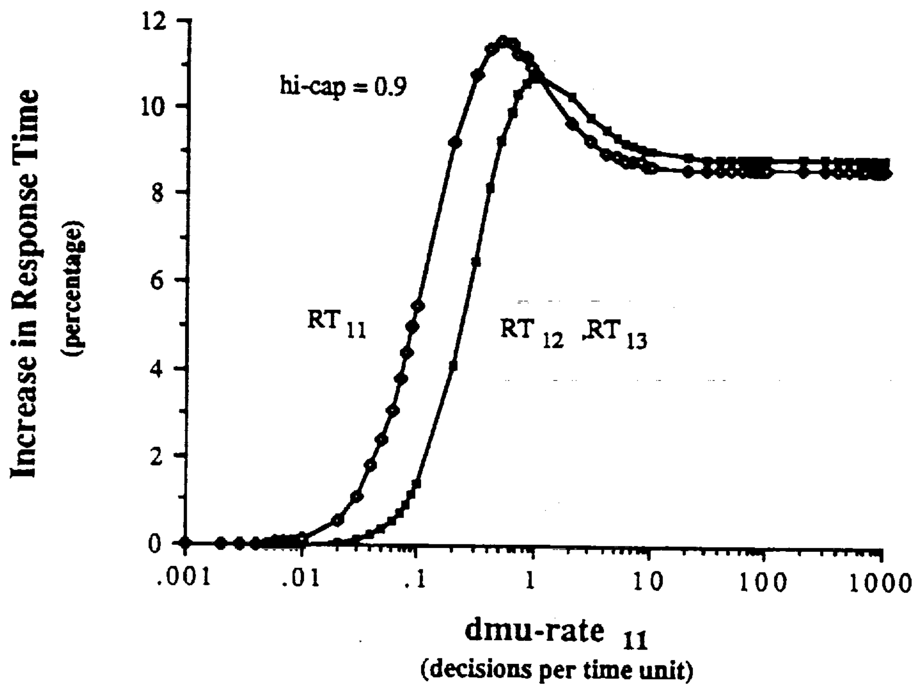


Figure 34: Percentage increase in response times of Figure 32 with hi-cap = 0.9.

9. PERFORMANCE EVALUATION OF HIER-3 DMA

The HIER-3 DMA of Figure 6 also controls manufacturing operations composed of three physical processes $\{PRO_1, PRO_2, PRO_3\}$. However, this time six computers are arranged in a three-level hierarchy. This could represent three workstations grouped in two cells, and supervised by a centralized shop controller. The modeler must choose the appropriate RANDOM or FIFO model that best suits the physical system at hand.

9.1 RANDOM HIER-3 DMA

The GSPN model in Figure 35 has a Reachability Graph with 13818 markings, and an Embedded Markov Chain with 5790 states. There are two random switches connected to OUT_{21} and OUT_{31}, and seven basic P-invariants. The response times are computed using Little's Law.

9.1.1 Response Time vs. Decision Making Rate

Figure 36 shows the effects of varying the Decision Making Rate, dmu-rate$_{11}$, on RT_{11}, RT_{12} and RT_{13}. The shape of the response is similar to that of the STAR-3 DMA but with $RT_{12} \neq RT_{13}$. The actual values are higher due to the increase in the number of DMUs that process a decision, however, the linear relationship found in previous models still holds. Note how increasing dmu-rate$_{11}$ has more influence on RT_{13} than on RT_{12}, which reflects a greater dependence on dmu-rate$_{11}$ due to the RANDOM behavior; a fact unobserved without the above modeling.

9.1.2 Response Time vs. Degree of Autonomy

Figure 37 shows the effects of varying the Degree of Autonomy, hi-aut$_{11}$, on the response time for three values of the Decision Making Capacity, hi-cap$_{11}$. Again, changing hi-aut$_{11}$ has more impact on the response times than changing

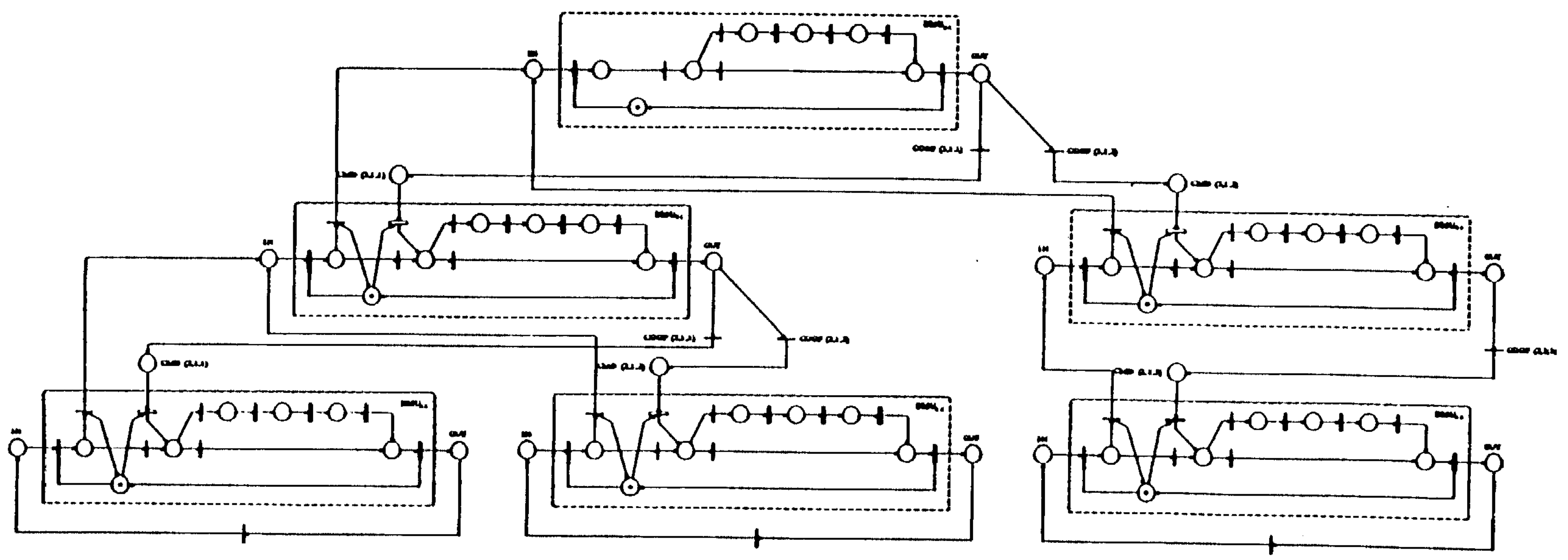


Figure 35: GSPN model of a RANDOM HIER-3 DMA.

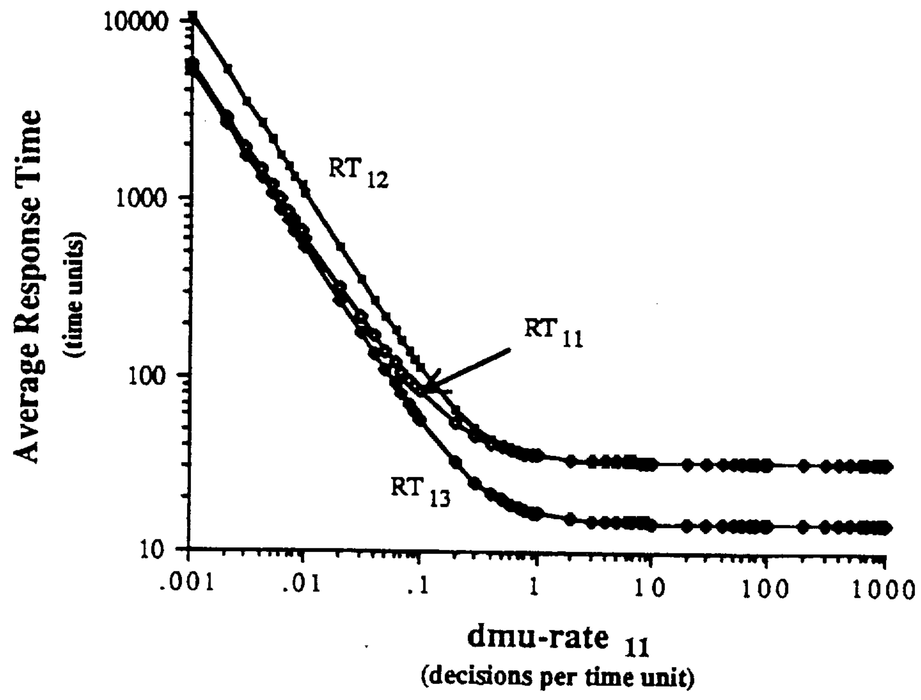


Figure 36: Response times vs. dmu-rate$_{11}$ of a RANDOM HIER-3 DMA.

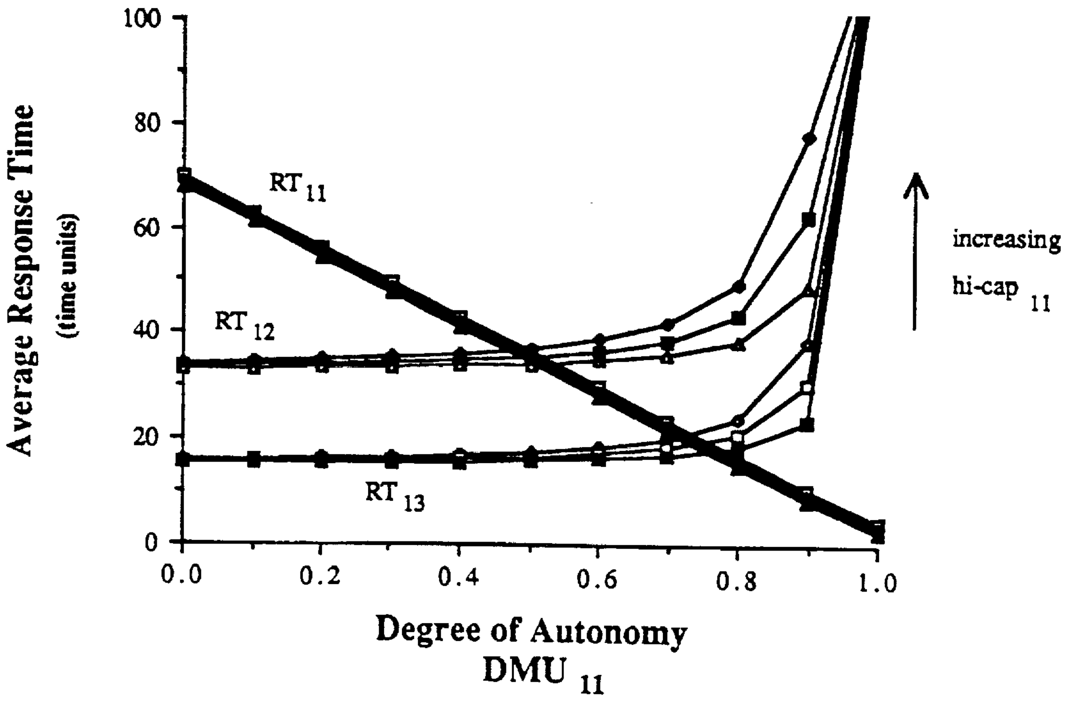


Figure 37: Response times vs. hi-aut$_{11}$ of a RANDOM HIER-3 DMA.

hi-cap$_{11}$. Increasing hi-aut$_{11}$ causes RT$_{11}$ to decrease linearly, and RT$_{12}$ and RT$_{13}$ to increase for the same reasons explained in section 8.2. Increasing hi-cap$_{11}$ also results in larger response times.

The effects of changing hi-aut and hi-cap of the other lower- and middle-level DMUs are shown in Figures 38-40. The linear relation between the response time of a DMU and its Degree of Autonomy still holds as for the STAR-3 DMA.

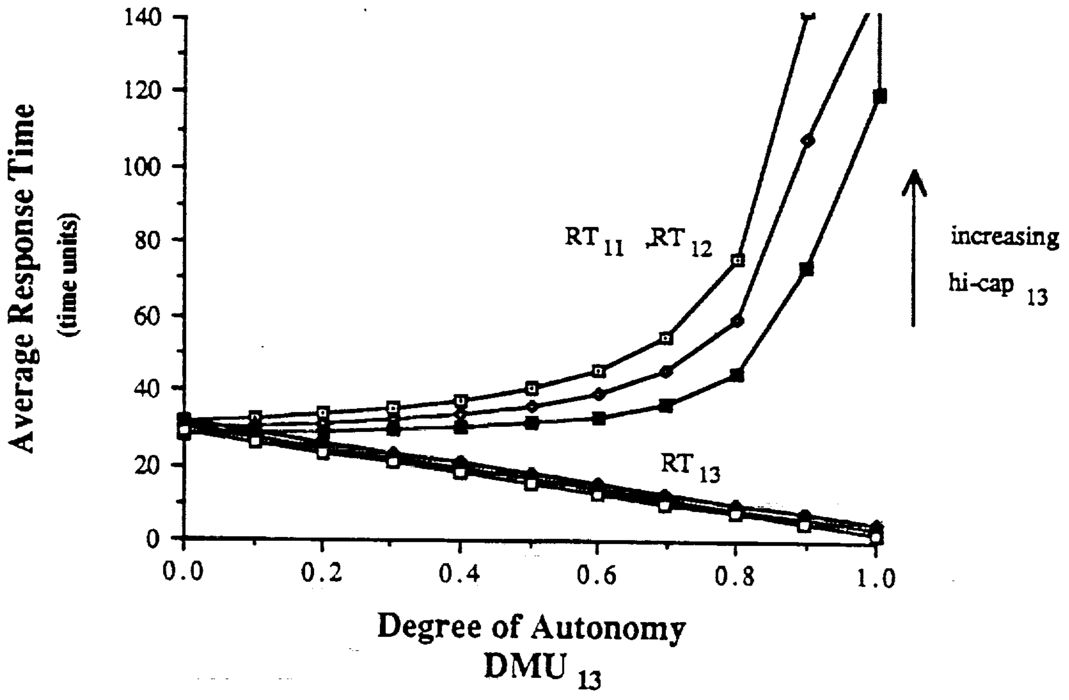


Figure 38: Response times vs. hi-aut$_{13}$ of a RANDOM HIER-3 DMA.

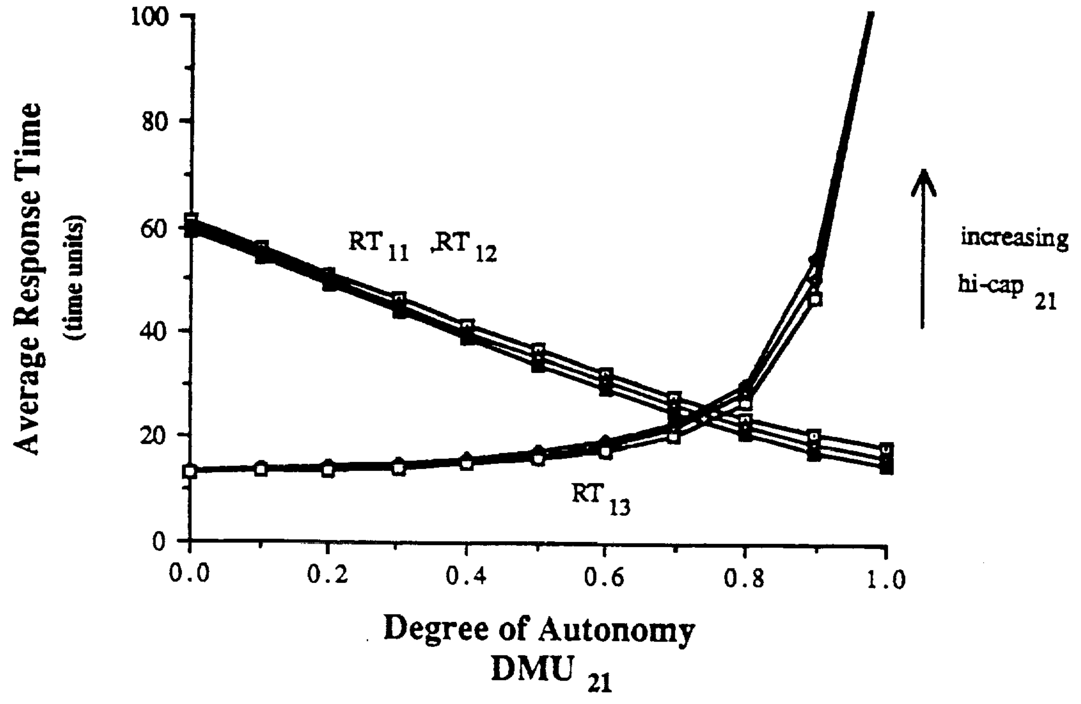


Figure 39: Response times vs. hi-aut$_{21}$ of a RANDOM HIER-3 DMA.

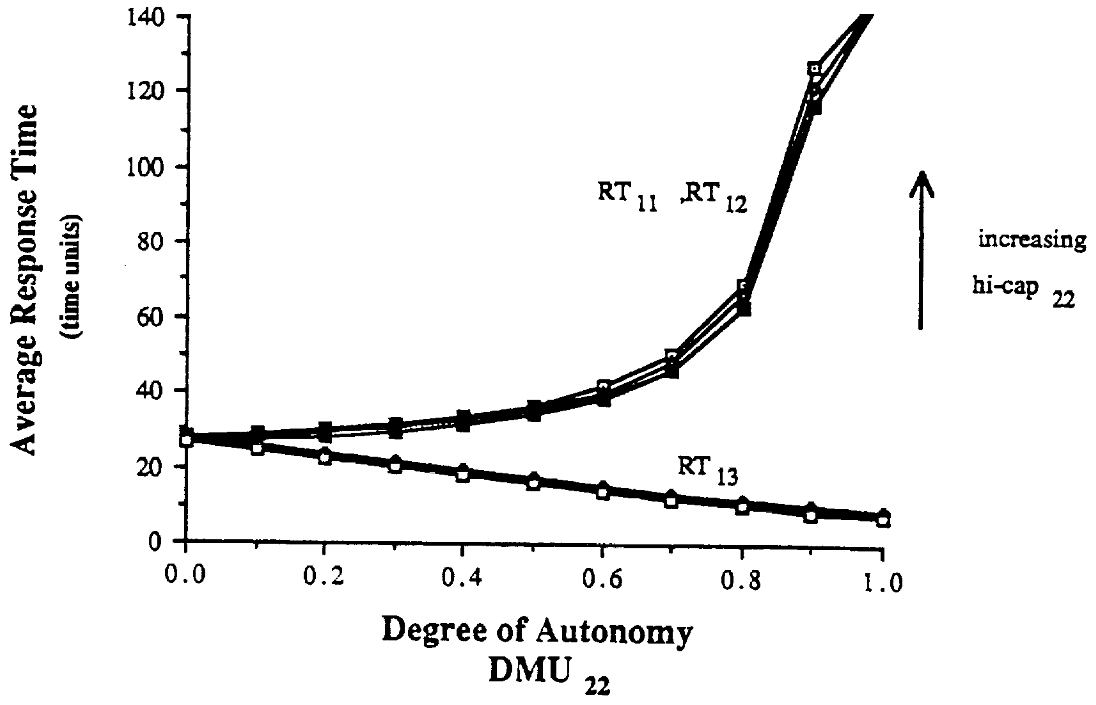


Figure 40: Response times vs. hi-aut$_{22}$ of a RANDOM HIER-3 DMA.

9.2 FIFO HIER-3 DMA

Combining the control logic of Figure 41, obtained by using the algorithm in the Appendix, with the GSPN model of Figure 35 results in a GSPN model with a FIFO behavior. The Reachability Graph has 14516 markings, and the Embedded Markov Chain has 5870 states.

9.2.1 Response Time vs. Decision Making Rate

From Figure 42 one can make observations similar to those of the FIFO STAR-3 DMA. In this case, however, $RT_{12} \neq RT_{13}$.

9.2.2 Response Time vs. Degree of Autonomy

The effects of changing the Degree of Autonomy, hi-aut, of each of the DMUs is shown in Figures 43-46 for three values of the Decision Making Capacity, hi-cap. The linear relationships discussed previously seem to hold in this case too. Also, the effect on RT_{12} and RT_{13} as hi-aut_{11} increases is negligible due to the decentralized nature of the FIFO DMA.

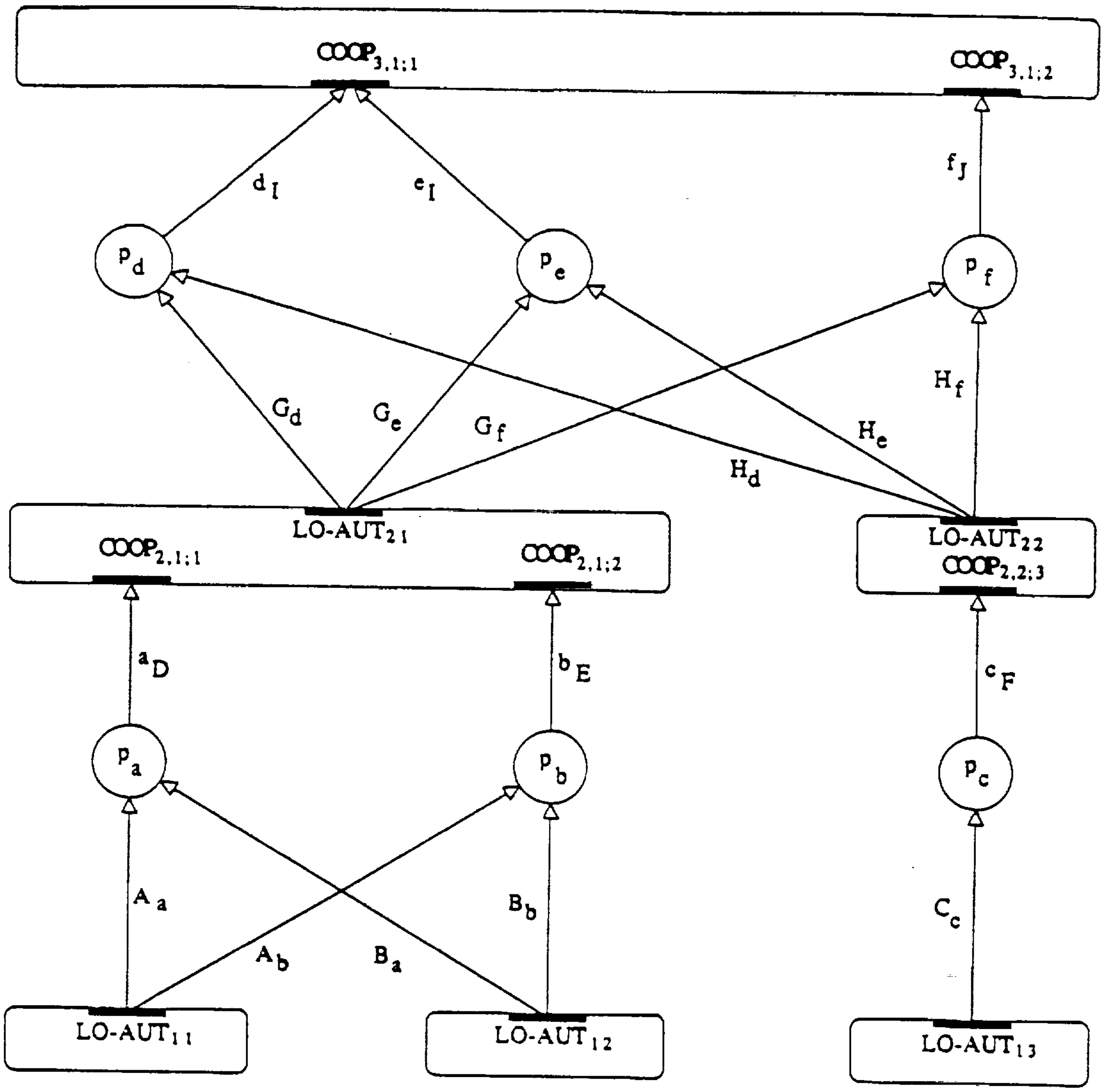


Figure 41: Control logic for FIFO HIER-3 DMA.

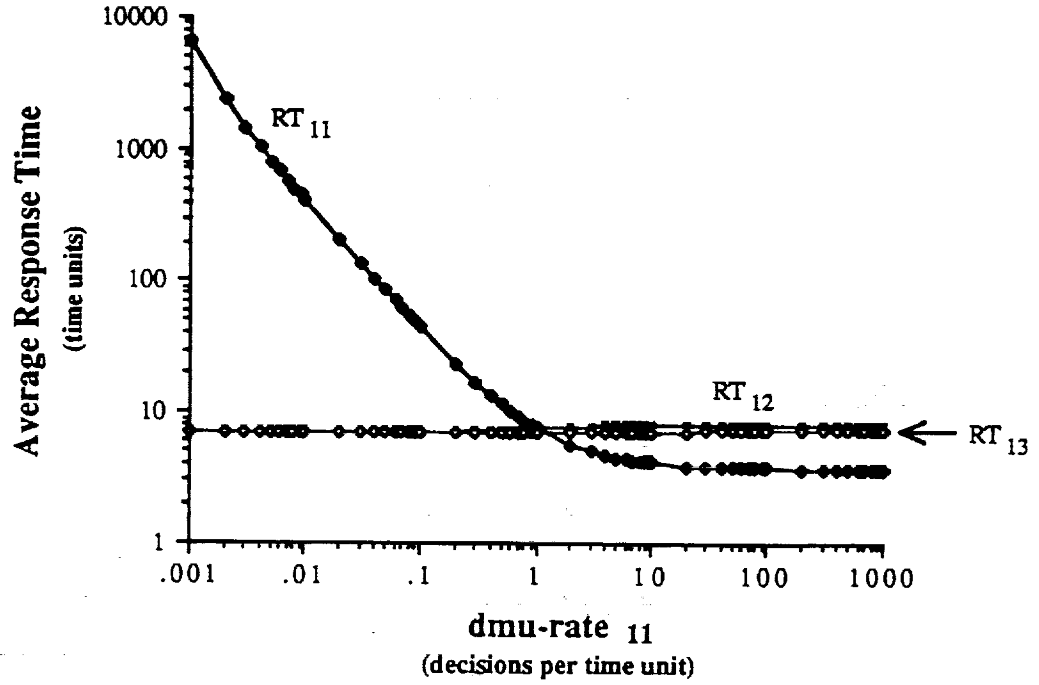


Figure 42: Response times vs. dmu-rate$_{11}$ of a FIFO HIER-3 DMA.

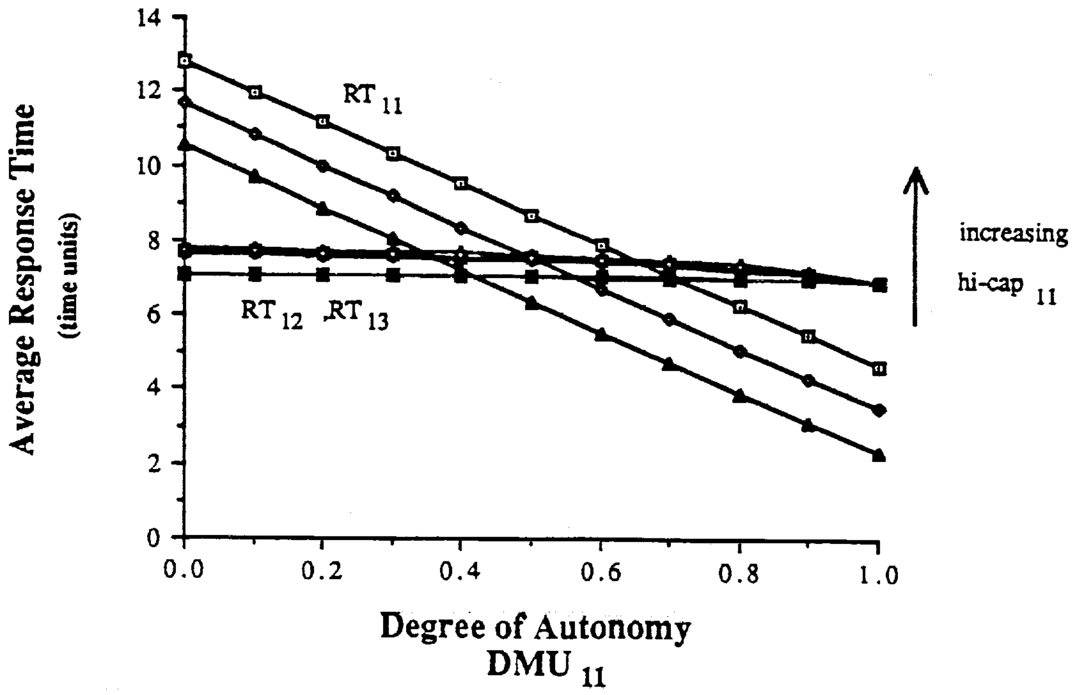


Figure 43: Response times vs. hi-aut$_{11}$ of a FIFO HIER-3 DMA.

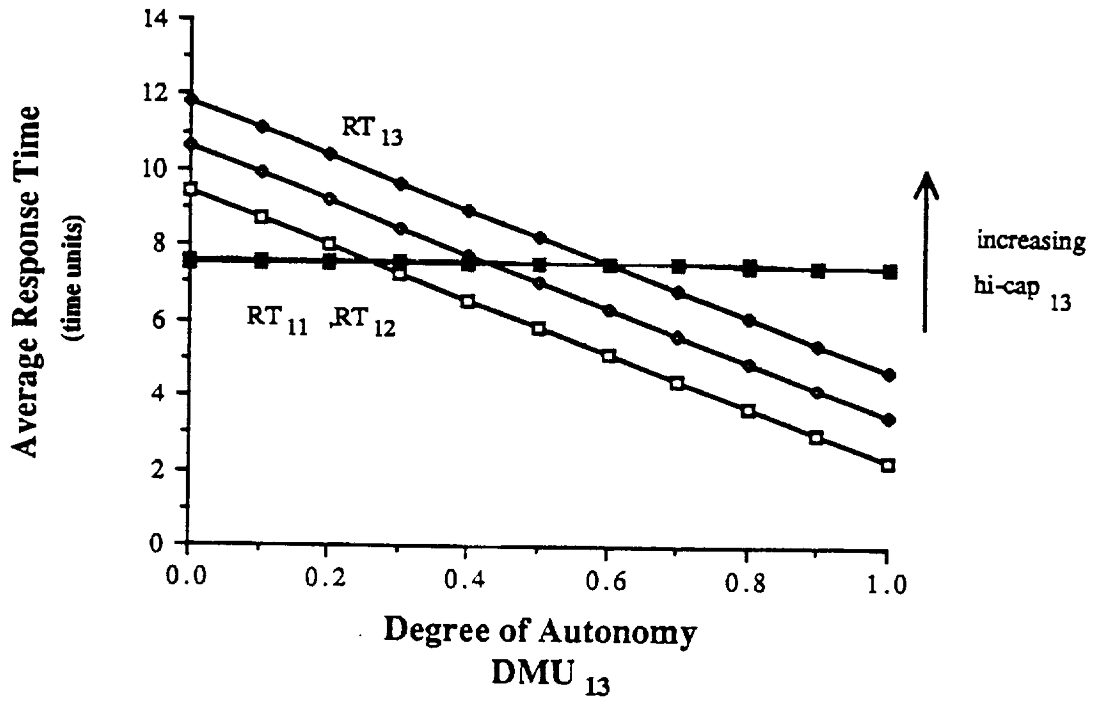


Figure 44: Response times vs. hi-aut$_{13}$ of a FIFO HIER-3 DMA.

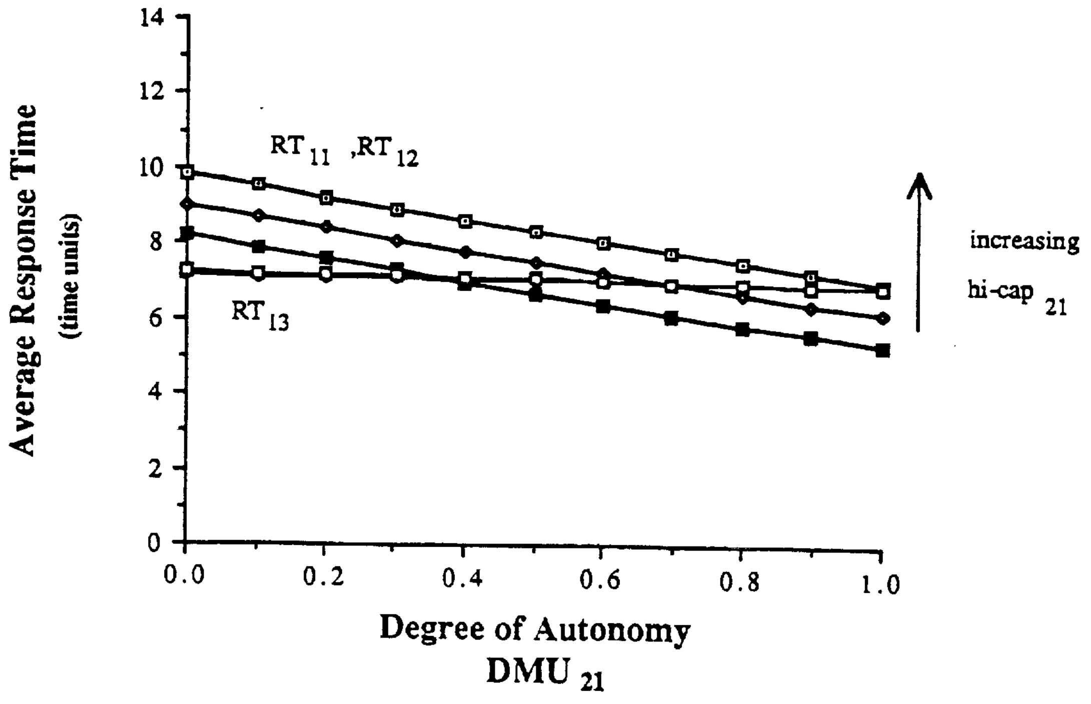


Figure 45: Response times vs. hi-aut$_{21}$ of a FIFO HIER-3 DMA.

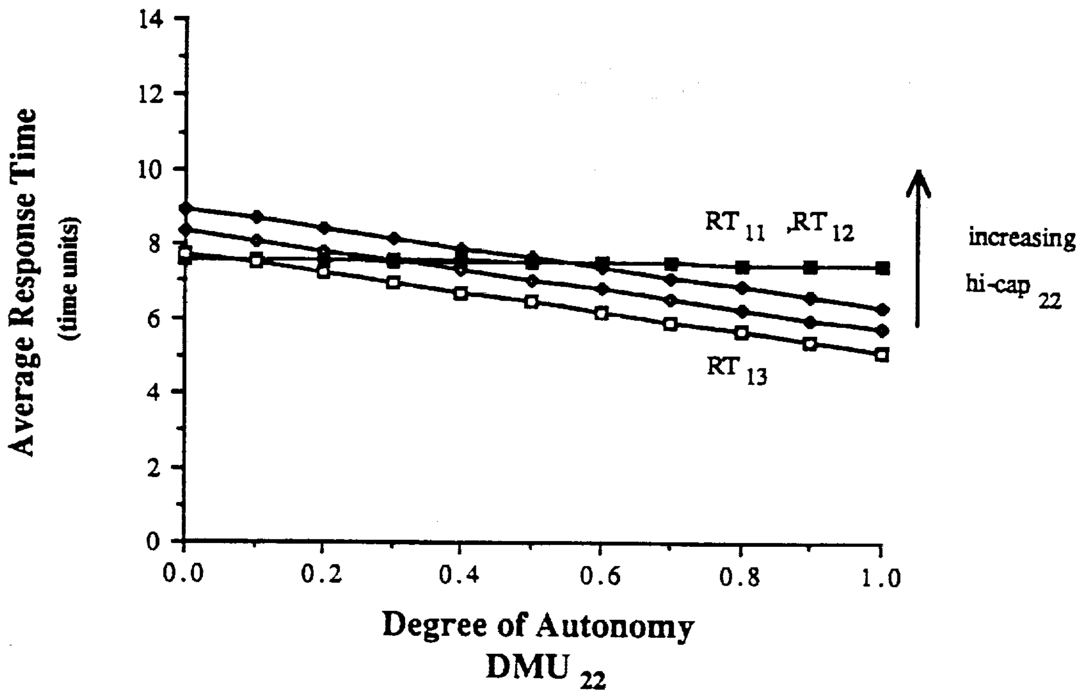


Figure 46: Response times vs. hi-aut$_{22}$ of a FIFO HIER-3 DMA.

10. DESIGNING DECISION MAKING ARCHITECTURES

The proposed methodology can be easily used as a design tool. Tradeoffs among alternative candidate DMAs can be evaluated to determine the most suitable DMA. The following examples illustrate this.

10.1 Example #1: Design a RANDOM HIER-3 DMA

In a RANDOM STAR-3 DMA, responding to the occurrence of 10 out of 100 potential events requires database accesses and information fusion algorithms. It is observed that, on average, the lower-level DMUs process a request in one time unit, and the upper-level DMU in ten time units. It is desired to decrease the response times by decreasing the load on DMU_{21}. Is there a RANDOM HIER-3 DMA that performs better subject to the condition that no changes can be done on the existing DMUs, and that the Decision Making Rate and Degree of Autonomy of the two middle-level DMUs must be equal?

This design problem requires the addition of two middle-level DMUs while making use of existing DMUs. The GSPN model with a Decision Making Capacity of hi-cap = 0.1 for all DMUs, dmu-rate_{21} = 0.1, and dmu-rate = 1 for the lower-level DMUs, gives a response time of 282.494 for the three low-level DMUs. The designer has the freedom to choose the rate and hi-aut of the two middle-level DMUs. Figure 47 shows the response time of a RANDOM HIER-3 DMA as the Degree of Autonomy changes for three dmu-rates, namely, 0.1, 1 and 10. If the dmu-rate = 0.1, then hi-aut = 0.78 is adequate. This implies that the two middle-level DMUs should locally handle at least 78 of the 100 potential events. For dmu-rate $\geq$ 1, hi-aut = 0.25 will be sufficient. Using faster DMUs, the desired performance can be achieved if the two middle-level DMUs can locally handle at least 25 of the 100 events.

Given that the existing DMA has a FIFO behavior with the same numerical values as above, can a RANDOM HIER-3 DMA be designed to have smaller response times? The FIFO STAR-3 GSPN model gives a response time of 94.503 for the three low-level DMUs. Figure 47 shows that hi-aut = 0.78 and dmu-rate $\geq$ 1 must be chosen. For dmu-rate $\leq$ 0.1 all the response times are not acceptable regardless to the

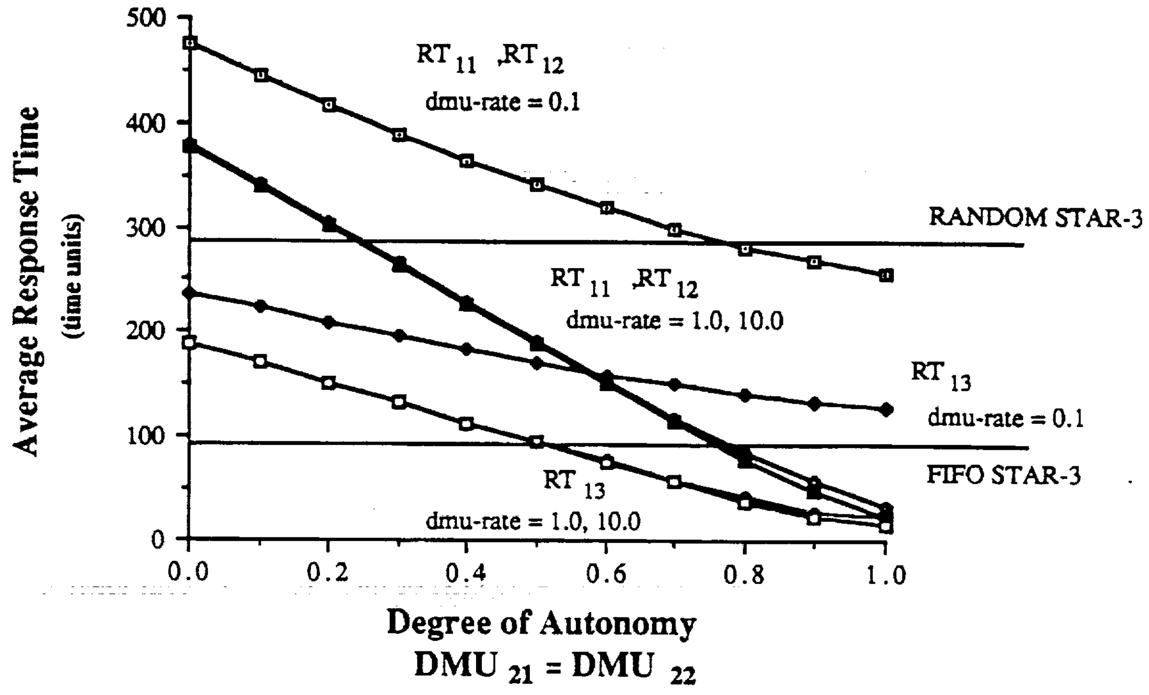


Figure 47: Response times for designing a RANDOM HIER-3 DMA.

Degree of Autonomy.

10.2 Example #2: Design a FIFO HIER-3 DMA

Given the same information as above, can a FIFO HIER-3 DMA be designed with smaller response times than the existing FIFO STAR-3 DMA? Figure 48 shows that choosing dmu-rate ≥ 1 will satisfy the desired performance, regardless of the Degree of Autonomy. This implies that the two additional middle-level DMUs need not process any requests locally. Choosing hi-aut = 0.0 will increase the cost and complexity without obtaining any improvements in the response times. However, Figure 48 also shows that dramatic improvements in the response times can be

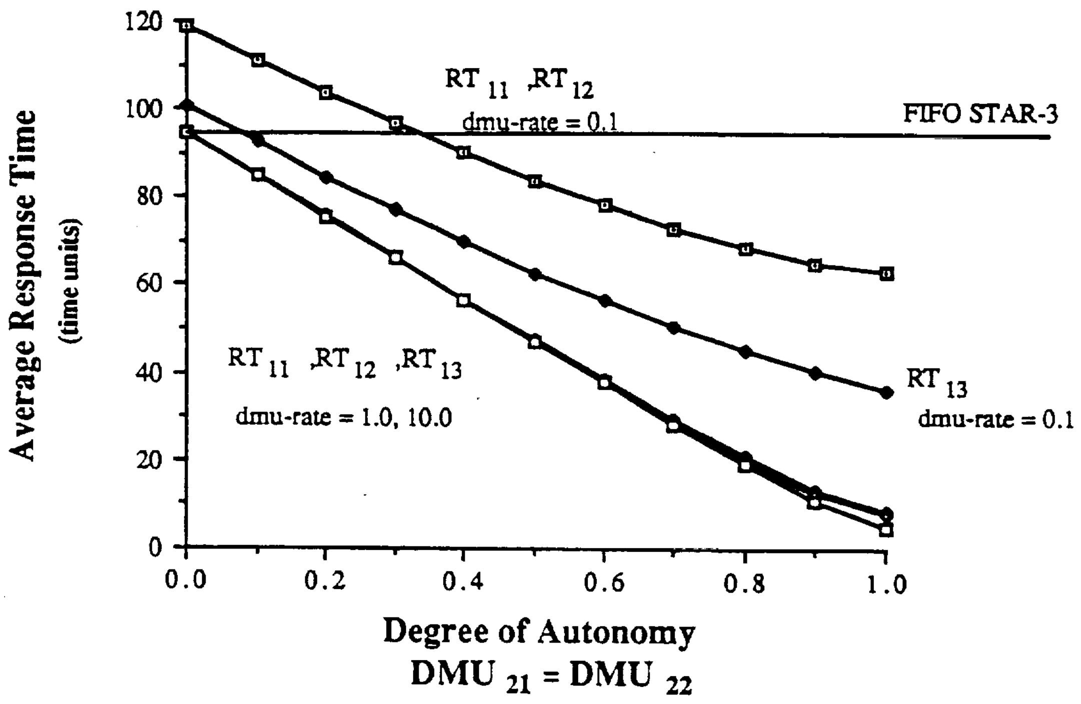


Figure 48: Response times for designing a FIFO HIER-3 DMA.

achieved if the Degree of Autonomy is increased. For dmu-rate = 0.1, hi-aut = 0.35 or greater is needed.

10.3 Example #3: Design a FIFO STAR-3 DMA

In this example, it is desired to reduce the cost and complexity of an existing FIFO HIER-3 DMA whose DMU_{22} can locally respond to 70 out of 100 potential events (i.e. $\text{hi-aut}_{22} = 0.7$). Can a FIFO STAR-3 DMA be designed subject to the condition that changes can be done only on the Degree of Autonomy of DMU_{11}?

This design problem requires the elimination of the two middle-level DMUs. To improve the performance, the average number of requests to DMU_{21} should be decreased. This can be achieved by having DMU_{11} to locally process as many events as needed. The FIFO HIER-3 GSPN model gives a response time of 6.977 for all DMUs. The designer has the freedom to choose only hi-aut_{11}. From Figure 49, hi-aut = 0.5 can be used to achieve the same performance with two less DMUs. In fact, further reduction in the response times can be obtained by making DMU_{11} process locally more than 50 out of the 100 events.

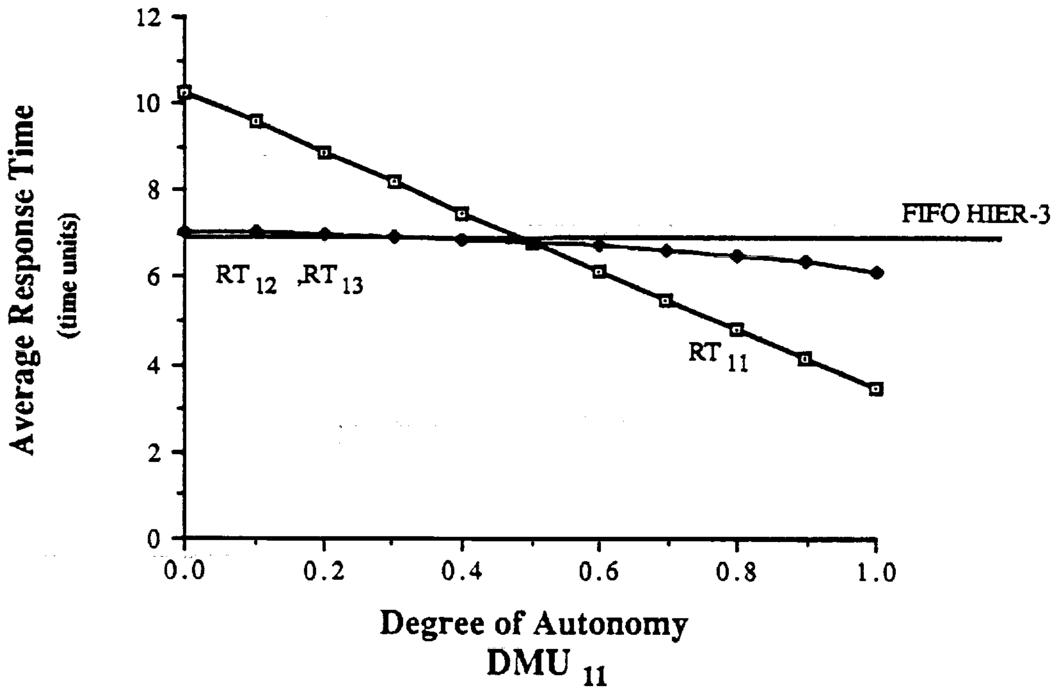


Figure 49: Response times for designing a FIFO STAR-3 DMA.

11. GSPN MODEL REDUCTION AND APPROXIMATION

Modeling CIM systems will undoubtedly result in large models. This is not a drawback of the modeling tool but is rather due to the size and complexity of CIM systems. The main advantage in using GSPNs over other methods such as Markov Chains is that they aid in managing this complexity. In this section, issues related to model reduction and approximation of GSPN models as they relate to the current work will be discussed. A brief discussion on the effects of the exponential assumption in GSPN models of several manufacturing systems can be found in [15].

The GSPN model of the FIFO HIER-3 DMA has a Reachability Graph with 14516 markings, and 29978 arcs. The Embedded Markov Chain has 5870 states with 35848 non-zero entries. Using a SUN4 system, SPNP [34] generated a solution in 10 minutes, while a VAX 750 was more than 20 times slower. Of course, the execution time depends on the type of processor used, amount of real memory, disk space and speed, and the number of logged-in users. Still, there will always be a limit on the size of models that can be analyzed.

Molloy has suggested the use of decomposition and aggregation techniques to deal with this complexity when using SPNs [22]. Decomposition methods operate on the Markov Chain description by manipulating the state transition matrix. Aggregation methods combine several SPNs into an approximate model.

11.1 Model Reduction

Obtaining a simpler GSPN model from another detailed GSPN model is called model reduction. A good approach to reducing the GSPN models of the transfer lines, production networks, and Decision Making Architectures would be to reduce the complexity, and hence the modeling details, of the basic GSPN modules.

Consider the isolated DMU in Figure 6. A simpler model, with the same steady-state behavior, and structural properties of the full model, is sought. With m(MP) = 1, the subnet between place CAP and the decision processing transition DP is safe and live. Let the desired reduced model for this subnet be as in Figure 50a. What should dmu-rate' be in order to have an equivalent steady-state probability distribution?

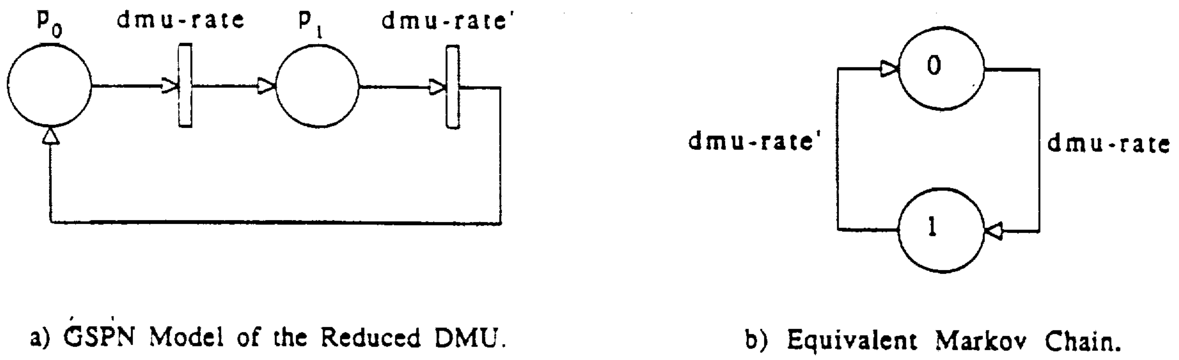

P_0
dmu-rate
P_1
dmu-rate'
0
dmu-rate'
dmu-rate
1
a) GSPN Model of the Reduced DMU.
b) Equivalent Markov Chain.

Figure 50: Equivalent GSPN model of part of an isolated DMU.

Figures 11b and 50b show the equivalent Markov Chain of the two models, respectively. The balance equations of the full model, with PRO being immediate, are

$$\begin{aligned}
\pi_0 \text{ dmu-rate} &= \pi_4 \text{ dmu-rate}, \\
\pi_2 \text{ dmu-rate} &= \pi_0 \text{ dmu-rate hi-cap}, \\
\pi_2 \text{ dmu-rate} &= \pi_1 \text{ dmu-rate}, \\
\pi_3 \text{ dmu-rate} &= \pi_2 \text{ dmu-rate}, \\
\pi_4 \text{ dmu-rate} &= \pi_3 \text{ dmu-rate} + \pi_0 \text{ dmu-rate lo-cap} = \pi_0 \text{ dmu-rate},
\end{aligned}$$

and for the reduced model is

$$\pi'_0 \text{ dmu-rate} = \pi'_1 \text{ dmu-rate}',$$

where the summation of all the probabilities in each case equals to 1.

Solving for π_0 and π'_0

$$\pi_0 = 1/(2 + 3 \text{ hi-cap}) \qquad (12),$$

$$\pi'_0 = 1/(1 + \text{dmu-rate}/\text{dmu-rate}') \qquad (13),$$

and equating π_0 and π'_0 gives

$$\text{dmu-rate}' = \text{dmu-rate}/(1 + 3 \text{ hi-cap}) = \text{dmu-rate}/(4 - 3 \text{ lo-cap}) \qquad (14).$$

Using this result the Reduced DMU is shown in Figure 51 which has three places, three timed and two immediate transitions instead of seven places, six timed and four immediate transitions. Note that for hi-cap =1, the rate of the equivalent transition is identical to that of four exponential servers in series, which form an Erlang distribution with parameter 4. One may also observe that using deterministic times, and for hi-cap = 1 and m(MP) = 1, the time delay of the equivalent transition is four times the time delay of the individual transitions.

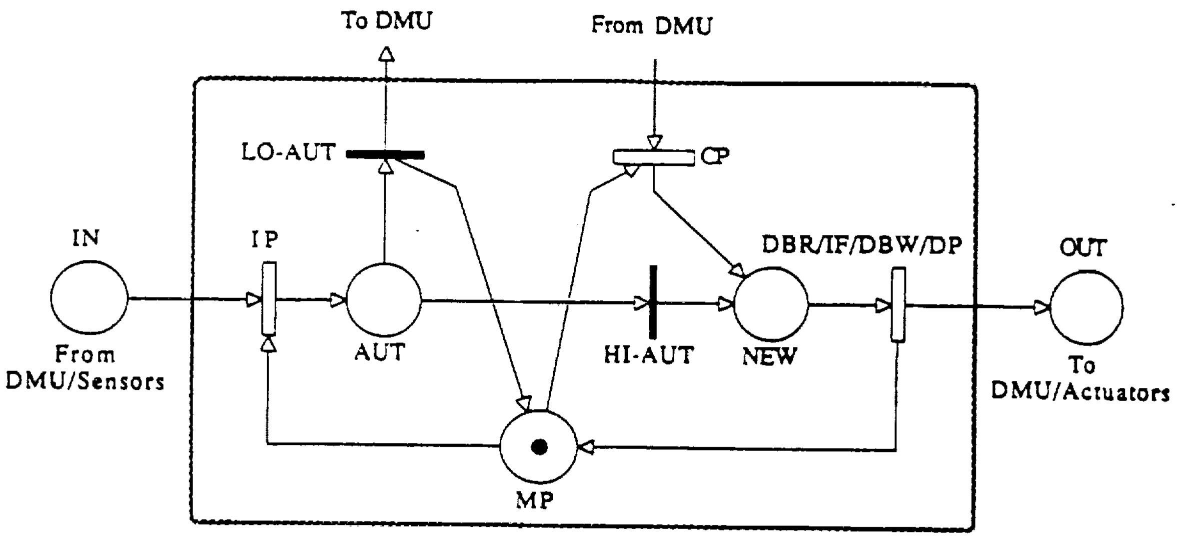


Figure 51: GSPN model of a Reduced Decision Making Unit.

Using the Reduced DMU, the isolated DMU and the STAR-1 DMA have the exact performance as with the full DMU model. Figures 52-53 compare the performance of the full and reduced models for the RANDOM and FIFO STAR-3 DMA. The RANDOM model has a Reachability Graph with 658 markings, and an Embedded Markov Chain with 242 states. The FIFO model has a Reachability Graph with 381 markings, and an Embedded Markov Chain with 129 states. The approximation for the RANDOM DMA is very good with a maximum error of -1%. For the FIFO DMA the error is almost within 1% except for a few points. Since the Reduced DMU was obtained using a safe DMU model it is expected that the performance of the reduced RANDOM STAR-3 DMA with m(MP) = 3 will not perform well. Figure 54 shows that the magnitude of the maximum error for this case is 51% and its magnitude is always greater than 15%. For this case, the GSPN model has a Reachability Graph with 814 markings, and an Embedded Markov Chain with 286 states. Note that all of the above reduced models exhibited a six- to eight-fold reduction in the state-space over their full-size counterparts, which translates into very significant reductions in the computation times. On a VAX 750, the full model takes more than 40 minutes, while the reduced model runs in under one minute.

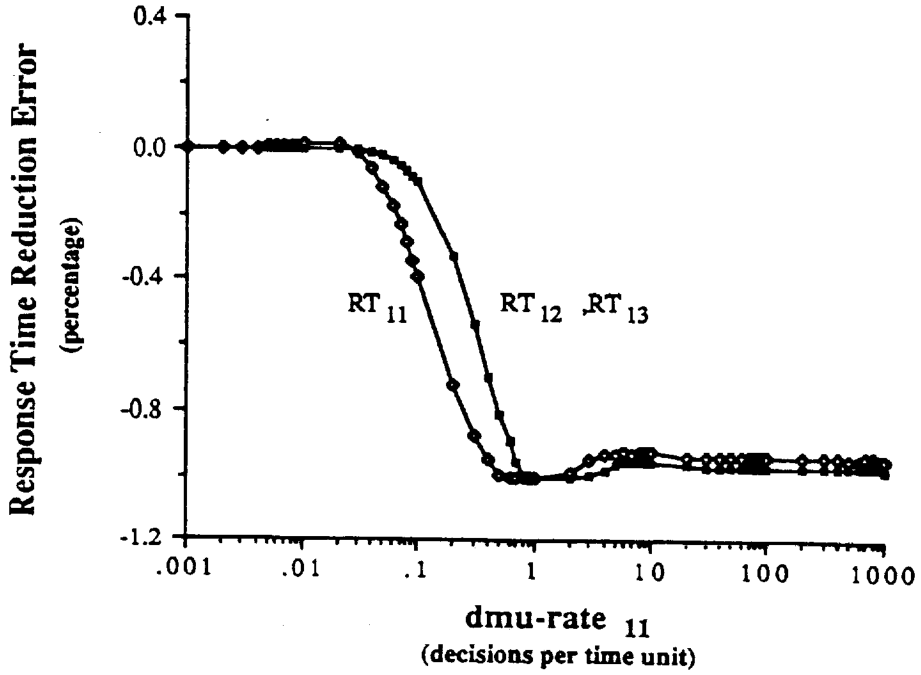


Figure 52: Reduction error of a RANDOM STAR-3 DMA.

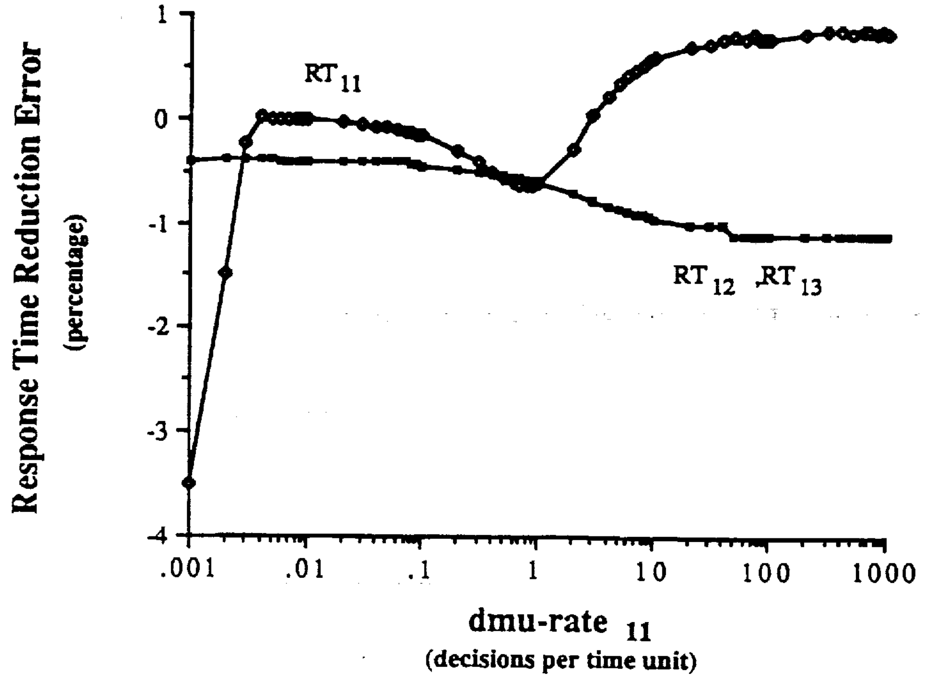


Figure 53: Reduction error of a FIFO STAR-3 DMA.

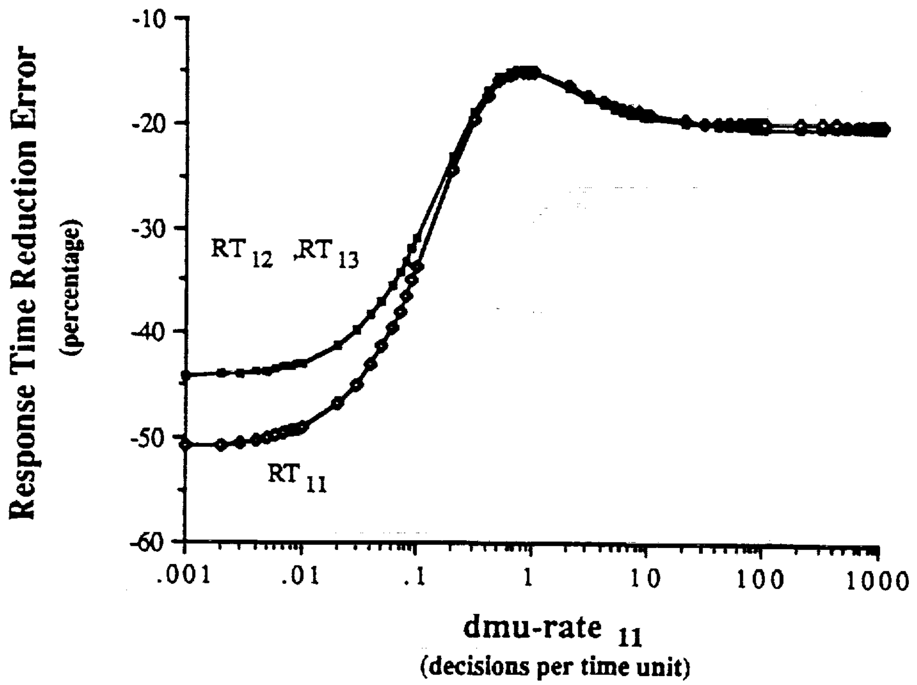


Figure 54: Reduction error of a RANDOM STAR-3 DMA with m(MP) = 3.

11.2 Model Approximation

Another approach to simplifying GSPN models is by model approximation. During the development of a model, the modeler judiciously uses some approximations in order to have a manageable final model. This is done without any analysis of the Reachability Graph or the equivalent Markov Chain. For example, if some event is known to be not too critical for the performance of the system, it can be eliminated. Also, if the rates of some timed transitions are orders of magnitude apart, then the faster ones could be replaced by immediate transitions. This will reduce the number of states in the Embedded Markov Chain, and by using efficient reachability graph generation algorithms such as those suggested in [28], the generation of the reachability graph will require less time and storage space.

12. SUMMARY OF THE RESULTS

The proposed methodology was successfully used to design and evaluate the performance of several real-time Decision Making Architectures for CIM systems. The functions and information requirements of a typical node in a manufacturing system were quantified using a GSPN model of a Decision Making Unit. The DMU served as a basic module from which DMA models were constructed, evaluated, and compared. The following conclusions can be drawn concerning the performance of the analyzed DMAs:

1. The response time is linearly dependent on the decision making time (or

inversely dependent on the Decision Making Rate), the Degree of Autonomy, and the Decision Making Capacity. It also depends on the actual, as well as the relative values of these variables.

2. A rule of thumb in the design of hierarchical control systems states that "errors must be identified and resolved at the lowest level possible" [2]. This work confirms that the response time could be more effectively reduced by increasing the Degree of Autonomy of the low-level DMUs.
3. For the analyzed DMAs, the response time was less sensitive to the Decision Making Capacity than the Degree of Autonomy. However, this is not necessarily the case in all situations.
4. The response times of more complex DMAs has the same profile as those of the simple STAR-1 DMA. This suggests that studying simpler models might give valuable insights into more complex models that are much harder to analyze.
5. It is not necessarily true that increasing the number of levels in a hierarchy results in larger response times. Speed, the Degree of Autonomy, and Decision Making Capacity all influence the response times. It was shown that a three-level hierarchical DMA could have a faster response time than a two-level one.
6. The addition of multiprocessing capability reduces the response times. Using a centralized database introduces some coupling among the DMUs which increases the response times.
7. Generally, RANDOM DMAs have a larger response time than FIFO DMAs. That is, it takes a longer time to get a response if interactions with other than parent DMUs are required.

13. CONCLUSIONS AND SUGGESTIONS FOR FUTURE RESEARCH

Basing the design of increasingly complex CIM systems solely on experience and intuition will undoubtedly result in inefficient and costly systems with unpredictable performances. Modeling is essential to ensure the orderly design, implementation, and control of CIM systems, and to gain valuable insights into their behavior. Results from these models can also be used as a guide to develop efficient data collection strategies for the experimental verification of these systems.

The present research shows that the proposed graph-based methodology is powerful and easy-to-use. It provides for a structured framework that enables the development of unified models for the performance evaluation of complex manufacturing systems. The proposed methodology was successfully used to evaluate and compare the performance of several real-time Decision Making Architectures. The topological choices of real-time DMAs for manufacturing systems were examined, and their effects on the response times were analyzed. The main emphasis was on the control aspects of these DMAs which determines the logical flow of the information; database issues were examined only briefly. The use of the methodology as a design tool was also demonstrated.

Several areas of research are still needed to achieve a fully integrated model for CIM systems:

1. **Database Models.** Databases are a major and crucial component in manufacturing systems. In this work, only one explicit but simple model of

a common database was analyzed to demonstrate how complex database models can be easily integrated with the models developed here. Detailed models of the hardware interactions as well as the needed database management protocols must be developed.

2. **Communication Network Models.** Just as with databases networking is a very important component of a CIM system. Models of various communication network topologies, such as star, ring and bus Local Area Networks, must be developed and integrated with the previous models.

3. **Model Reduction and Approximation.** When modeling complex manufacturing systems using PNs, the modeler should attempt to make judicious choices concerning the details to be included in the model. This is needed to ensure that the resulting models are tractable. GSPNs provide a quick and easy-to-use modeling tool for evaluating complex CIM systems. Upon gaining some understanding of the system's behavior, one can resort to other tools such as simulation to get a more accurate analysis. Some ideas on model reduction and approximation were presented in section 11. It is desirable to identify classes of GSPN models to which model reduction algorithms can be applied without the need to generate the reachability graph. Such an approach appears to be feasible for live and safe nets, but further investigation is needed. Extending that to more general models is a greater challenge. It would also be beneficial to develop methods for estimating the reduction error.

4. **Improvements on existing PN software.** Using Colored GSPNs to

evaluate the performance of DMAs would be very valuable. Colors can be used to distinguish the originator of the requests and the type of processing needed. Also, one may assign different probabilities to the random switches depending on the colors of the tokens. Often one evaluates the same model as a particular variable (transition rate) changes. Therefore, significant computational savings can be obtained by adding the capability of storing the equivalent Markov Chain and computing the performance measures for different transition rates without having to generate the Reachability Graph more than once. Finally, incorporating Perturbation Analysis [13] techniques into PN packages is desirable.

5. **Modeling the Manufacturing Process.** In this research, the manufacturing process was modeled as an immediate transition PRO. This was done in order to gain better insights into the behavior of DMAs without adding the complexities of the physical system. It is desirable to evaluate alternative DMAs for a specific process, where PRO can be replaced by the appropriate GSPN model. Models of the actual decision making process, and of the database transactions might be required.

6. **Integrating Modeling Methodologies.** GSPNs are probably the best suited tools to develop integrated models for manufacturing systems. Other modeling techniques such as simulation, Queueing Networks and Markov Chains, have several features that complement GSPNs. In order to develop the most efficient and flexible modeling tool, several of these techniques should be integrated in one software package that includes the features discussed in section 2. For example, a graphical interface such as that

provided with GreatSPN [35] can be used to generate the GSPN description for the SPNP [34] package. The same graphical interface can also be used to develop some complex FIFO model, while the temporal analysis could be done using Queueing Networks if the model has a product-form solution.

7. **Control Logic**. Algorithms for the synthesis of supervisory control logic are needed to ensure the proper operation of CIM systems. The algorithm in the Appendix, which implements a FIFO net using GSPNs, is one example. Petri Nets [36] and Formal languages [14] were suggested as possible models. A potential approach is to describe the system's behavior using P- and T-invariants, and synchronic distances [37] from which the control logic can be synthesized.

8. **Distributed Decision Making.** The real-time DMAs analyzed in this paper had a hierarchical functional decomposition. They are distributed in the sense that several loosely connected but geographically separate computers have access to distributed databases. The actual decision making process was not distributed. Evaluating the performance of decision making algorithms based on Distributed Artificial Intelligence such as Contract Nets [38-39] would be valuable.

Integrating control, database, and communication models could be advantageous when analyzing CIM systems. That was the motivation for suggesting the use of a DMU as a basis for modeling and evaluating real-time DMAs. Although detailed database models have been evaluated [40-43], architectures of multiprocessor systems were studied [44], and several Local Area Network models were investigated [45], none have been done in the context of integrated manufacturing. The modeling issues are different when developing integrated models for manufacturing. It is not clear at this stage what level of abstraction should these models include. Is there a substantial advantage in using an integrated model over separate models? Can one obtain meaningful performance estimates by using aggregate but separate models instead of a larger detailed model? Answering these questions is a great challenge that remains to be met.

"One never notices what has been done; one can only see what remains to be done..."

Marie Curie

14. ACKNOWLEDGMENTS

We wish to thank Dr. Gianfranco Ciardo for the SPNP package that was used to obtain all the numerical results in this work. The support of the CIM program of Rensselaer Polytechnic Institute's Center for Manufacturing Productivity and Technology Transfer is greatly appreciated.

15. REFERENCES

[1] National Research Council: "A Research Agenda for CIM, Information Technology". National Academy Press, Washington, D. C., 1988.

[2] Johnson, T. L.: "Distributed Hierarchical Control Architectures for Automation". Computer Architectures for Robotics and Automation, Graham, J. H. (ed.), Gordon and Breach Science Publishers, New York, 1987, pp. 173-194.

[3] March, J. G. and Simon, H. A.: Organizations. Wiley, New York, 1958.

[4] Tabak, D. and Levis, A. H.: "Petri Net Representation of Decision Models". IEEE Transactions on Systems, Man, and Cybernetics, volume SMC-15, number 6, November/December 1985, pp. 812-818.

[5] Hillion, H. P.: "Performance Evaluation of Decisionmaking Organizations using Timed Petri Nets". M. Sc. Thesis LIDS-TH-1590, MIT, August, 1986.

[6] Remy, P. A., Levis, A. H. and Jin, V. Y.-Y.: "On the Design of Distributed Organizational Structures". Automatica, volume 24, number 1, January 1988, pp. 81-86.

[7] Andreadakis, S. K. and Levis, A. H.: "Design Methodology for Command and Control Organizations". Report LIDS-P-1681, MIT, July, 1987.

[8] Scott, H. A., Davis, R. P., Wysk, R. A. and Nunnally, C. E.: "Hierarchical Control Model for Automated Manufacturing Systems". Computer and Industry, volume 7, number 3, Fall 1983, pp. 241-255.

[9] Desrochers, A. A.: Modeling and Control of Automated Manufacturing Systems. To be published by the IEEE Computer Society Press, 1989.

[10] Bel, G., and Dubois, D.: "A Unified Setting for Various Models of Automated Material Flow and Production Systems". Computers in Industry, volume 6, 1985, pp. 227-235.

[11] Dubois, D.: "Flexible Manufacturing Systems: Modelling and Simulation".

The Encyclopedia of Systems and Control, Singh, M. (ed.), Pergamon Press, 1987.

[12] Suri, R.: "An Overview of Evaluative Models for Flexible Manufacturing Systems". Annals of Operations Research, volume 3, 1985.

[13] Ho, Y. C.: "Performance Evaluation and Perturbation Analysis of Discrete Event Dynamic Systems". IEEE Transactions on Automatic Control, volume AC-32, number 7, July 1987, pp. 563-572.

[14] Wonham, W. M.: "A Control Theory for Discrete-Event Systems". Systems Control Group Report #8714, Department of Electrical Engineering, University of Toronto, Toronto, Canada, December 1987.

[15] Al-Jaar, R. Y.: "Performance Evaluation of Automated Manufacturing Systems using Generalized Stochastic Petri Nets". Ph. D. Thesis, Rensselaer Polytechnic Institute, Troy, NY, March 1989.

[16] Al-Jaar, R. Y. and Desrochers, A. A.: "Petri Nets in Automation and Manufacturing". To appear in Advances in Automation and Robotics, Saridis, G. N. (ed.), JAI Press Inc., volume 2, Connecticut, 1989.

[17] Al-Jaar, R. Y. and Desrochers, A. A.: "A Survey of Petri Nets in Automated Manufacturing Systems". Proceedings of the 12th IMACS World Congress, Paris, France, July 18-22, 1988, pp. 503-510.

[18] Al-Jaar, R. Y. and Desrochers, A. A.: "A Modular Approach for the Performance Evaluation of Automated Manufacturing Systems using Generalized Stochastic Petri Nets". Submitted to the IEEE Transactions on Robotics and Automation.

[19] Al-Jaar, R. Y. Desrochers, A. A. and DiCesare, F.: "Evaluation of Part-Type Mix for a Machining Workstation using Generalized Stochastic Petri Nets". Proceedings of the IEEE Conference on Decision and Control, Austin Texas, December 7-9, 1988, pp. 2307-2313.

[20] Al-Jaar, R. Y. and Desrochers, A. A.: "Modeling and Analysis of Transfer Lines and Production Networks using Generalized Stochastic Petri Nets". Proceedings of the Conference on University Programs in Computer Aided Engineering, Design and Manufacturing, Atlanta, Georgia, June 27-29, 1988, pp. 12-21.

[21] Zhou, M., DiCesare, F. and Desrochers, A. A.: "A Top-Down Approach to Systematic Synthesis of Petri Net Models for Manufacturing Systems". To appear in the Proceedings of the IEEE Robotics and Automation Conference, Scottsdale, Arizona, May 1989.

[22] Molloy, M. K.: "Performance Analysis using Stochastic Petri Nets". IEEE Transactions on Computers, volume C31, number 9, September 1982, pp. 913-917.

[23] Natkin, S. O.: "Les réseaux de Petri stochastiques et leur application à l'évaluation des systèmes informatiques". Thèse de Docteur-Ingénieur, Conservatoire National des Arts et Métiers (CNAM), Paris, June 1980.

[24] Ajmone Marsan, M., Balbo, G. and Conte, G.: "A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems". ACM Transactions on Computer Systems, volume 2, number 2, May 1984, pp. 93-122.

[25] Crockett, D., Desrochers, A. A., DiCesare, F. and Ward, T.: "Implementation of a Petri Net Controller for a Machining Workstation". Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, North Carolina, March 30-April 3 1987, pp. 1861-1867.

[26] O'Grady, P. J.: Controlling Automated Manufacturing Systems. Kogan Page Ltd, 1986.

[27] Ajmone Marsan, M., Balbo, G., Chiola, G. and Conte, G.: "Generalized Stochastic Petri Nets Revisited: Random Switches and Priorities". Proceedings of the International Workshop on Petri Nets and Performance Models, Madison, Wisconsin, August 24-26, 1987, pp. 44-53.

[28] Balbo, G., Chiola, G., Franceschinis, G. and Molinar Roet, G.: "On the Efficient Construction of the Tangible Reachability Graph of Generalized Stochastic Petri Nets". Proceedings of the International Workshop on Petri Nets and Performance Models, Madison, Wisconsin, August 24-26, 1987, pp. 136-145.

[29] Simpson, J., Hocken, R. and Albus, J.: "The automated manufacturing research facility of the National Bureau of Standards". Journal of Manufacturing Systems, volume 1, number 1, 1982, pp. 17-31.

[30] Trivedi, K. S.: Probability & Statistics with Reliability, Queuing, and Computer Science Applications. Prentice-Hall, 1982.

[31] Roucairol, G.: "FIFO Nets". Petri Nets: Central Models and Their Properties, Brauer, W., Reisig, W. and Rozenberg, G. (eds.), Springer-Verlag, New York, 1987, pp. 436-459.

[32] Zenie, A.: "Colored Stochastic Petri Nets". Proceedings of the IEEE International Workshop on Timed Petri Nets, Torino, Italy, July 1-3, 1985, pp. 262-271.

[33] Gentina, J. C. and Corbeel, D.:"Coloured Adaptive Structured Petri Net: A Tool for the Automatic Synthesis of Hierarchical Control of Flexible Manufacturing Systems". Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, North Carolina, March 30-April 3, 1987, pp. 1166-1173.

[34] Ciardo, G.: "Manual for the SPNP Package". February 1989.

[35] Chiola, G.: "A Graphical Petri Net Tool for Performance Analysis". Proceedings of the 3rd International Workshop on Modeling Techniques and Performance Evaluation, AFCET, Paris, France, March 1987.

[36] Krogh, B. H.: "Controlled Petri Nets and Maximally Permissive Feedback Logic". Proceedings of the Twenty-Fifth Annual Allerton Conference on Communication, Control, and Computing, Monticello, Illinois, September 30-October 2, 1987, pp. 317-326.

[37] Silva, M.: "Towards a Synchrony Theory for P/T Nets". Concurrency and Nets, Voss, K., Genrich, H. J., Rozenberg, G. (eds.), Springer-Verlag, Berlin, 1987, pp. 435-460.

[38] Van Dyke Parunak, H.: "Manufacturing Experience with the Contract Net". Distributed Artificial Intelligence, Huhns, M. N. (ed.), Pitman, London, 1987, pp. 285-310.

[39] Shaw, M. J. and Whinston, A. B.: "Task Bidding and Distributed Planning in Flexible Manufacturing". Proceedings of the IEEE Second Conference on AI Applications, Miami, Florida, December, 1985, pp. 184-189.

[40] Genrich, H. J.: "Predicate/Transition Nets". Petri Nets: Central Models and Their Properties, Brauer, W., Reisig, W. and Rozenberg, G. (eds.), Springer-Verlag, New York, 1987, pp. 207-247.

[41] Voss, K.: "Nets in Databases". Petri Nets: Central Models and Their Properties, Brauer, W., Reisig, W. and Rozenberg, G. (eds.), Springer-Verlag, New York, 1987, pp. 97-134.

[42] Hura, G. S., Singh, H. and Nanda, N. K.: "Some Design Aspects of Databases Through Petri Net Modeling". IEEE Transactions on Software Engineering, volume SE-12, number 4, April 1986, pp. 505-510.

[43] Dugan, J. B. and Ciardo, G.: "Stochastic Petri Net Analysis of a Replicated File System". Proceedings of the International Workshop on Petri Nets and Performance Models, Madison, Wisconsin, August 24-26, 1987, pp. 84-92.

[44] Ajmone Marsan, M., Chiola, G. and Conte, G.: "Stochastic Petri Nets as a Tool for the Analysis of High Performance Distributed Architectures". Supercomputing Systems, Kartashev, L. and S. (eds.), New York: Van Nostrand, 1987.

[45] Ajmone Marsan, M. and Signore, V.: "Timed Petri Net Performance Models of Fiber Optics LAN Architectures". Proceedings of the International Workshop on Petri Nets and Performance Models, Madison, Wisconsin, August 24-26, 1987, pp. 66-74.

16. APPENDIX: ALGORITHM FOR SYNTHESIS OF FIFO MODELS

The following algorithm allows for the synthesis of FIFO models for a Decision Making Architecture, regardless of the number of levels. Of course, the availability of software packages for FIFO nets or Colored GSPNs would not require the use of such an algorithm. Therefore, the present algorithm can be seen as a technique for emulating the behavior of FIFO nets and Colored GSPNs in the absence of the appropriate software tools.

The following algorithm applies to any system where several processes share one or more resources in a FIFO structure. For the DMAs, the processes are the individual DMUs that issue a request for intervention, and the shared resource is the parent DMU that processes these requests. Note that since every segment of the physical manufacturing process is assigned an exclusive DMU, the transitions PRO are ignored. The algorithm will be presented using the generic term process, instead of DMU.

Algorithm

This algorithm keeps track of the order of occurrence of the requests by assigning a "control" place to each and every process that issues a request. The process whose control place has the largest number of tokens in a given marking is at the top of the queue. This "book-keeping" is done by depositing and removing the appropriate number of tokens in the control places using marking-dependent arcs, and enabling functions [33-34]. To achieve this, the following definitions are needed:

$SR(\Pi_K)$ ≡ {all transitions that Send a FIFO Request to process Π_K}.

$RA(\Pi_K)$ ≡ {all transitions that Receive an Answer from process Π_K}.

$AP(\Pi_K)$ ≡ {all control places that input from transitions in $SR(\Pi_K)$}. These places are Answer Places.

$IN(X)$ ≡ {all control places whose one and only one output transition is t_X}. These places input to t_X.

$OUT(Y)$ ≡ {all control places that belong to the process containing transition t_Y, and that input from t_Y}. Transition t_Y outputs to these places.

Then,

1. Create a control place for each of the lowest level processes.
2. Create the same number of control places at all other levels, if applicable.
3. Connect every transition in $SR(\Pi_K)$ to every control place of the processes with a transition belonging to $SR(\Pi_K)$.
4. Connect every place in $AP(\Pi_K)$ to its <u>respective</u> transition in $RA(\Pi_K)$.
5. The weight of the arc from transition t_L to place p_i is defined as

follows:

$$L_i = \begin{cases} 1, & m(p_i) > 0 \text{ or } p_i \text{ is the First Leftmost Empty} \\ & \text{Place (FLEP) in OUT(L).} \\ 0, & \text{else.} \end{cases}$$

6. For all transitions t_L in RA(Π_K), the weight of the arc from place p_i to transition t_L is defined as follows:

$$i_L = \begin{cases} m(p_i), & m(p_i) = \max\{m(p_j)\} \\ & \text{for } p_j \in AP(\Pi_K). \\ 0, & \text{else.} \end{cases}$$

7. To every transition t_L in RA(Π_K), assign an enabling function E_L defined as follows:

$$E_L = \begin{cases} 1, & \max\{m(p_i)\} = \max\{m(p_j)\} \\ & \text{for } p_i \in IN(L),\ p_j \in AP(\Pi_K). \\ 0, & \text{else.} \end{cases}$$

The enabling function guarantees that at most one transition in RA(Π_K) is enabled in a given marking.

As an illustration, the above algorithm will be used to implement the FIFO control logic for the STAR-3 and HIER-3 DMAs.

Control Logic for the FIFO STAR-3 DMA

Following Figure 25, the following sets can be identified:

$SR(DMU_{11}) = SR(DMU_{12}) = SR(DMU_{13}) = \phi.$

$SR(DMU_{21}) = \{LO\text{-}AUT_{11}, LO\text{-}AUT_{12}, LO\text{-}AUT_{13}\}.$

$RA(DMU_{11}) = RA(DMU_{12}) = RA(DMU_{13}) = \phi.$

$RA(DMU_{21}) = \{COOP_{2,1;1}, COOP_{2,1;2}, COOP_{2,1;3}\}.$

$AP(DMU_{11}) = AP(DMU_{12}) = AP(DMU_{13}) = \phi.$

$AP(DMU_{21}) = \{p_a, p_b, p_c\}.$

$IN(LO\text{-}AUT_{11}) = IN(LO\text{-}AUT_{12}) = IN(LO\text{-}AUT_{13}) = \phi.$

$IN(COOP_{2,1;1}) = \{p_a\}$, $IN(COOP_{2,1;2}) = \{p_b\}$, $IN(COOP_{2,1;3}) = \{p_c\}$.

$OUT(LO\text{-}AUT_{11}) = \{p_a\}$, $OUT(LO\text{-}AUT_{12}) = \{p_b\}$, $OUT(LO\text{-}AUT_{13}) = \{p_c\}$.

$OUT(COOP_{2,1;1}) = OUT(COOP_{2,1;2}) = OUT(COOP_{2,1;3}) = \phi.$

The functions of the marking-dependent arcs are:

$$A_a = 1.$$

$$A_b = \begin{cases} 1, & m(p_b) > 0. \\ 0, & \text{else.} \end{cases}$$

$$A_c = \begin{cases} 1, & m(p_c) > 0. \\ 0, & \text{else.} \end{cases}$$

$$B_a = \begin{cases} 1, & m(p_a) > 0. \\ 0, & \text{else.} \end{cases}$$

$$B_b = 1.$$

$$B_c = \begin{cases} 1, & m(p_c) > 0. \\ 0, & \text{else.} \end{cases}$$

$$C_a = \begin{cases} 1, & m(p_a) > 0. \\ 0, & \text{else.} \end{cases}$$

$$C_b = \begin{cases} 1, & m(p_b) > 0. \\ 0, & \text{else.} \end{cases}$$

$$C_c = 1.$$

$$a_D = \begin{cases} m(p_a), & m(p_a) = \max\{m(p_a), m(p_b), m(p_c)\}. \\ 0, & \text{else.} \end{cases}$$

$$b_E = \begin{cases} m(p_b), & m(p_b) = \max\{m(p_a), m(p_b), m(p_c)\}. \\ 0, & \text{else.} \end{cases}$$

$$c_F = \begin{cases} m(p_c), & m(p_c) = \max\{m(p_a), m(p_b), m(p_c)\}. \\ 0, & \text{else.} \end{cases}$$

The enabling functions are:

$$E_D = \begin{cases} 1, & m(p_a) = \max\{m(p_a), m(p_b), m(p_c)\}. \\ 0, & \text{else.} \end{cases}$$

$$E_E = \begin{cases} 1, & m(p_b) = \max\{m(p_a), m(p_b), m(p_c)\}. \\ 0, & \text{else.} \end{cases}$$

$$E_F = \begin{cases} 1, & m(p_c) = \max\{m(p_a), m(p_b), m(p_c)\}. \\ 0, & \text{else.} \end{cases}$$

Control Logic for the FIFO HIER-3 DMA

Following Figure 41, the following sets can be identified:

$SR(DMU_{11}) = SR(DMU_{12}) = SR(DMU_{13}) = \phi.$

$SR(DMU_{21}) = \{LO\text{-}AUT_{11}, LO\text{-}AUT_{12}\}$, $SR(DMU_{22}) = \{LO\text{-}AUT_{13}\}$,

$SR(DMU_{31}) = \{LO\text{-}AUT_{21}, LO\text{-}AUT_{22}\}$.

$RA(DMU_{11}) = RA(DMU_{12}) = RA(DMU_{13}) = \phi$.

$RA(DMU_{21}) = \{COOP_{2,1;1}, COOP_{2,1;2}\}$, $RA(DMU_{22}) = \{COOP_{2,2;3}\}$,

$RA(DMU_{31}) = \{COOP_{3,1;1}, COOP_{3,1;2}\}$.

$AP(DMU_{11}) = AP(DMU_{12}) = AP(DMU_{13}) = \phi$.

$AP(DMU_{21}) = \{p_a, p_b\}$, $AP(DMU_{22}) = \{p_c\}$, $AP(DMU_{31}) = \{p_d, p_e, p_f\}$.

$IN(LO\text{-}AUT_{11}) = IN(LO\text{-}AUT_{12}) = IN(LO\text{-}AUT_{13}) = \phi$.

$IN(COOP_{2,1;1}) = \{p_a\}$, $IN(COOP_{2,1;2}) = \{p_b\}$, $IN(COOP_{2,2;3}) = \{p_c\}$.

$IN(LO\text{-}AUT_{21}) = IN(LO\text{-}AUT_{22}) = \phi$.

$IN(COOP_{3,1;1}) = \{p_d, p_e\}$, $IN(COOP_{3,1;2}) = \{p_f\}$.

$OUT(LO\text{-}AUT_{11}) = \{p_a\}$, $OUT(LO\text{-}AUT_{12}) = \{p_b\}$, $OUT(LO\text{-}AUT_{13}) = \{p_c\}$.

$OUT(COOP_{2,1;1}) = OUT(COOP_{2,1;2}) = OUT(COOP_{2,2;3}) = \phi$.

$OUT(LO\text{-}AUT_{21}) = \{p_d, p_e\}$, $OUT(LO\text{-}AUT_{22}) = \{p_f\}$.

$OUT(COOP_{3,1;1}) = OUT(COOP_{3,1;2}) = \phi$.

The functions of the marking-dependent arcs are:

$$A_a = 1.$$

$$A_b = \begin{cases} 1, & m(p_b) > 0. \\ 0, & \text{else.} \end{cases}$$

$$B_a = \begin{cases} 1, & m(p_a) > 0. \\ 0, & \text{else.} \end{cases}$$

$$B_b = 1.$$

$$C_c = 1.$$

$$G_d = 1.$$

$$G_e = \begin{cases} 1, & m(p_e) > 0 \quad \text{or} \quad p_e \text{ is FLEP in OUT(LO-AUT}_{21}\text{).} \\ 0, & \text{else.} \end{cases}$$

$$G_f = \begin{cases} 1, & m(p_f) > 0. \\ 0, & \text{else.} \end{cases}$$

$$H_d = \begin{cases} 1, & m(p_d) > 0. \\ 0, & \text{else.} \end{cases}$$

$$H_e = \begin{cases} 1, & m(p_e) > 0. \\ 0, & \text{else.} \end{cases}$$

$$H_f = 1.$$

$$a_D = \begin{cases} m(p_a), & m(p_a) = \max\{m(p_a), m(p_b)\}. \\ 0, & \text{else.} \end{cases}$$

$$b_E = \begin{cases} m(p_b), & m(p_b) = \max\{m(p_a), m(p_b)\}. \\ 0, & \text{else.} \end{cases}$$

$$c_F = m(p_c).$$

$$d_I = \begin{cases} m(p_d), & m(p_d) = \max\{m(p_d), m(p_e), m(p_f)\}. \\ 0, & \text{else.} \end{cases}$$

$$e_I = \begin{cases} m(p_e), & m(p_e) = \max\{m(p_d), m(p_e), m(p_f)\}. \\ 0, & \text{else.} \end{cases}$$

$$f_J = \begin{cases} m(p_f), & m(p_f) = \max\{m(p_d), m(p_e), m(p_f)\}. \\ 0, & \text{else.} \end{cases}$$

The enabling functions are:

$$E_D = \begin{cases} 1, & m(p_a) = \max\{m(p_a), m(p_b)\}. \\ 0, & \text{else.} \end{cases}$$

$$E_E = \begin{cases} 1, & m(p_b) = \max\{m(p_a), m(p_b)\}. \\ 0, & \text{else.} \end{cases}$$

$$E_F = 1.$$

$$E_I = \begin{cases} 1, & \max\{m(p_d), m(p_e)\} = \max\{m(p_d), m(p_e), m(p_f)\}. \\ 0, & \text{else.} \end{cases}$$

$$E_J = \begin{cases} 1, & \max\{m(p_f)\} = \max\{m(p_d), m(p_e), m(p_f)\}. \\ 0, & \text{else.} \end{cases}$$